Raspberry PI Internet Radio

Constructors Manual

Bob Rathbone Computer Consultancy

www.bobrathbone.com

20th of May 2014

Version 3.5

Contents

Tables

Figures

Introduction

This manual describes how to create an Internet Radio using the Raspberry PI educational computer. The source and basic construction details are available from the following web site: http://www.bobrathbone.com/raspberrypi_radio.htm

This manual provides a more detailed overview of construction and software installation than provided by the web site. It contains instructions for building the radio using either the HDD44780 LCD directly wired to the Raspberry PI GPIO pins or using an Adafruit RGB-backlit LCD plate for Raspberry PI. It can be constructed using either push buttons or rotary encoders.

The features of the radio are:

- Raspberry PI running standard Music Player Daemon (MPD)
- Three different LCDs are supported
  - 2 x 16 character LCD with HD44780 controller
  - 4 x 20 character LCD with HD44780 controller
  - Adafruit LCD plate with 5 push buttons (I2C interface)
- Clock display or IP address display (for web interface)
- Five push button operation (Menu, Volume Up, Down, Channel Up, Down)
- As alternative to the above rotary encoder switches may be used
- Timer (Snooze) and Alarm functions
- Artist and track scrolling search function
- Plays music from a USB stick, SD card or from a Network drive (NAS)
- Menu option to display a single RSS news feed
- Web interface using snoopy and others
- Plays Radio streams or MP3 and WMA tracks
- Output either using the analogue audio jack or a USB speaker set.
- Play output on PC or on a mobile device using ICECAST streaming
- Playlist creation program using a list of URLs
- Controlled by an Object Orientated Python application
- Support for European character sets (Limited by LCD capabilities)

Various examples of the Raspberry PI internet radio were built using this design.

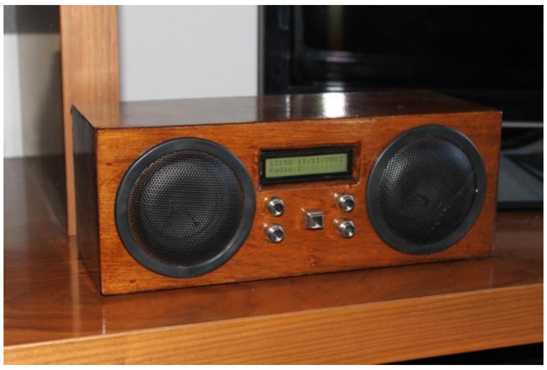


Figure 1 The completed classic style radio

This classic style Internet Radio is built into a wooden case. This is using two four inch speakers and audio amplifier stripped out from an old pair of PC speakers. It has five buttons in all. The centre square button is the menu selection.

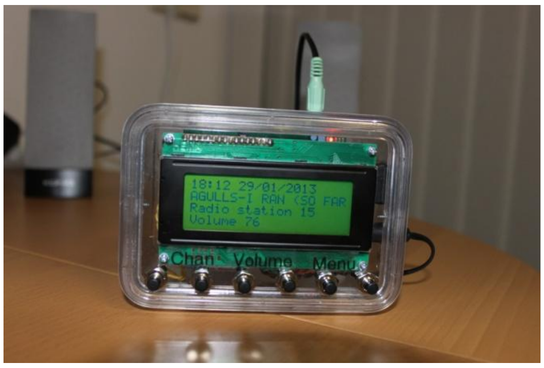


Figure 2 Four line LCD radio in a clear case

Example of the Internet Radio with a four line by twenty character compatible HD4478 LCD display. The transparent case was an old cream cracker box.

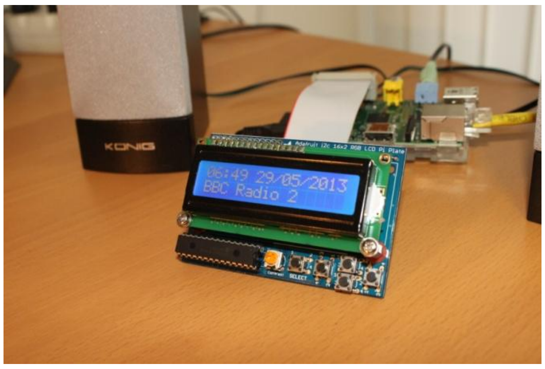


Figure 3 Radio using the Adafruit LCD plate

Example of the PI internet radio using an Adafruit RGB-backlit LCD plate for Raspberry PI from AdaFruit industries. It has five push buttons and is the easiest option to construct. If you want to build this into a case then don't use the buttons supplied with the kit but use external buttons.

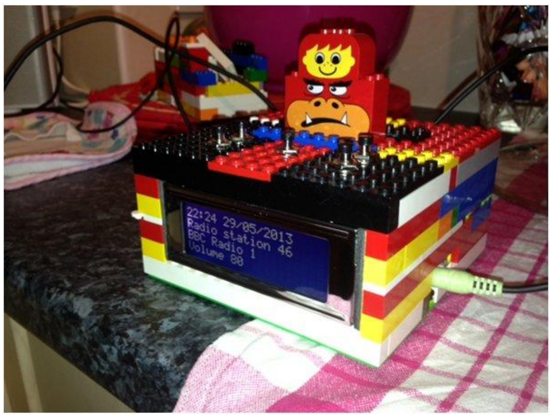


Figure 4 Lego internet radio

Example of a fun radio built using this design and Lego from Alan Broad (United Kingdom). This really puts the fun back into computing.

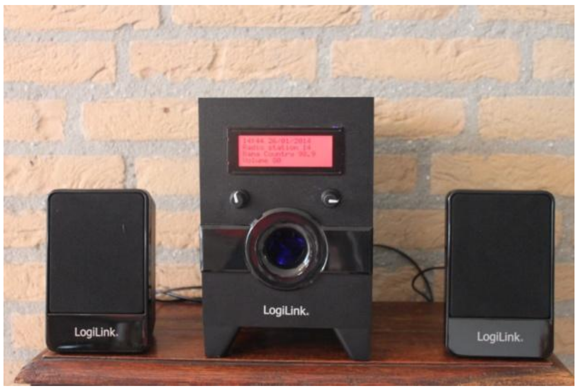


Figure 5 Pi radio using rotary encoders

The rotary encoder switch version of the radio consists of a Raspberry PI connected to an Adafruit 20 character x 4 line RGB LCD display housed. It is all housed in a LogiLink PC speaker set with two rotary encoder switches. The rotary encoders also have push buttons (Push the knob in). The left one is the *Mute* switch and the right one is the *Menu* switch. The blue glow in the sub-woofer opening comes from a bright blue LED.

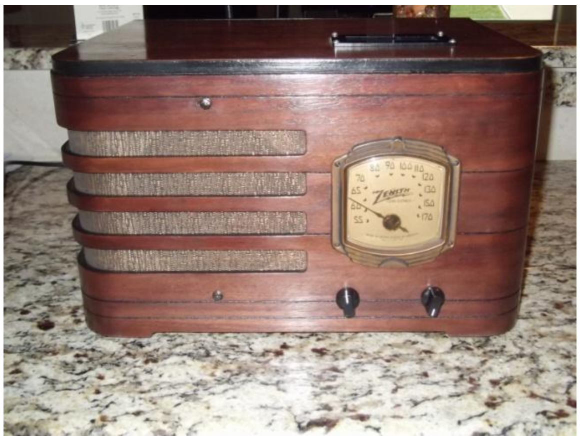


Figure 6 Old Zenith radio using rotary encoders

Example of the PI radio from James Rydell built into an old Zenith valve radio case. The pictures below show the inside and top view respectively. The two original controls have bee replaced by two rotary switches. The old valve radio inside has been completely removed and replaced with the Raspberry PI and radio components. The LCD display has been built into the top so as not to spoil the original face of the radio. This is a fine example of what can be done with this project.

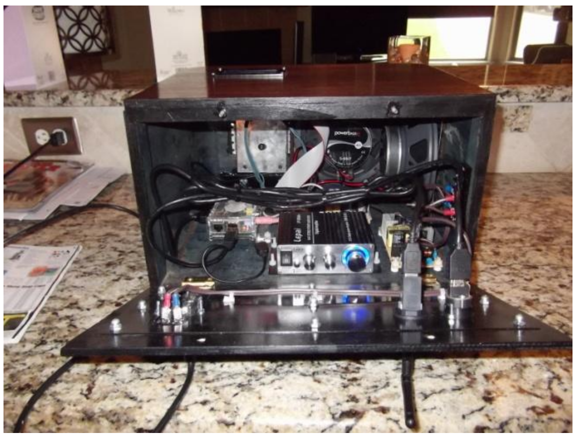


Figure 7 Zenith radio rear view

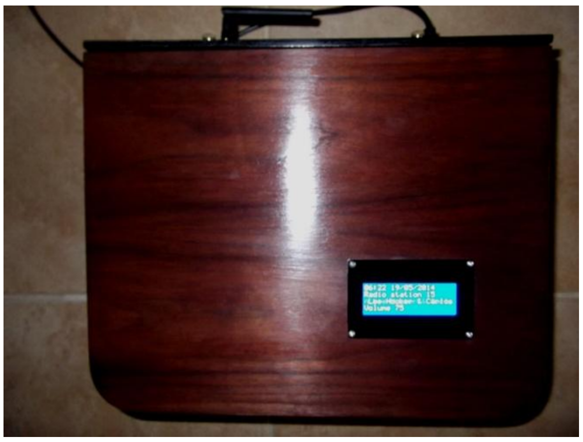


Figure 8 Zenith radio top view

There are many more examples of what people have done to house their PI radio but unfortunately they can’t all be shown in this manual.

For alternative ideas see both the next pages and the constructors page at http://www.bobrathbone.com/pi_radio_constructors.htm

Hardware

The principle hardware required to build the radio consists of the following components:

- A Raspberry PI computer
- An HD44780 LCD display or an Adafruit RGB-backlit LCD plate for the Raspberry PI
- LCD and switches interface board

Raspberry PI computer

The **Raspberry Pi** is a credit-card-sized single-board computer developed in the United Kingdom by the Raspberry Pi Foundation with the intention of promoting the teaching of basic computer science in schools.

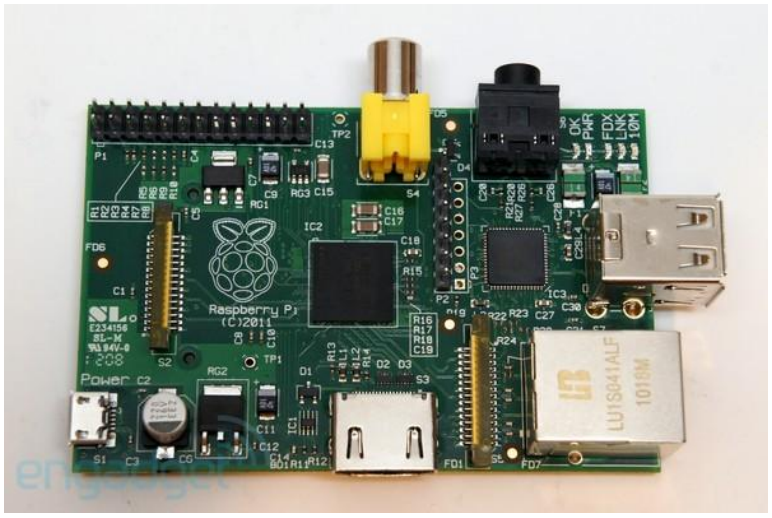


Figure 9 Raspberry PI Computer

More information on the Raspberry PI computer may be found here:
http://en.wikipedia.org/wiki/Raspberry_Pi

If you are new to the Raspberry PI try the following beginners guide. http://elinux.org/RPi_Beginners

The HD44780 LCD display

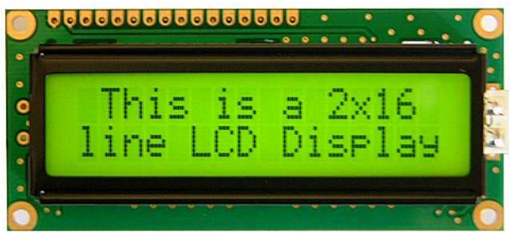


Figure 10 The HD44780 LCD display

The HDD44780 LCD interface is an industry standard interface for a variety of LCD displays. These can come in various sizes but the two lines by 16 character display is the most popular. The software for this Internet radio also supports the four lines by twenty character display. Most of the 16×2 modules available are compatible with the Hitachi HD44780 LCD controller so there is a wide choice of these displays. For pin-out details see *LCD* Module Wiring on page 15.

Radio variants

Before starting you need to make a choice which type of radio you are going to build. There are five possible versions that can be constructed as shown in the following table.

Table 1 Radio variants

Variant	Description	Display Type	Controls	Program
1	Two line 16 character LCD with push buttons	Two line LCD	Five push buttons	radiod.py
2	Four line 20 character LCD with push buttons	Four line LCD	Five push buttons	radio4.py
3	Two line 16 character LCD with rotary encoders	Two line LCD	Two rotary encoders with push buttons	rradiod.py
4	Four line LCD with rotary encoders	Four line LCD	Two rotary encoders with push buttons	rradio4.py
5	Adafruit LCD plate with push buttons	Two line LCD (via SPI interface)	Five push buttons (Via SPI interface)	ada_radio.py

All of these use a different program at present as shown in the last column of the above table.

Which one to use is a matter of personal choice. The AdaFruit LCD plate is without a doubt the easiest to construct. The others require more effort but are worth the trouble. It is then a simple choice of which display (two or four line) and whether to use rotary encoders or push button switches. The rotary encoder options give the most natural feel for the radio as most conventional radios use knobs to control the volume and station tuning. The four line LCD can display more information.

Housing the radio

This manual describes a couple of ways of housing the radio. A few ideas are below:

- A custom built case as shown in this manual
- Old plastic boxes or food containers
- Construct a case using Lego
- Use a pair of speaker housings that have enough room
- Install in an old vintage radio (really cool)
- Use an old wooden wine box
- Use an old video recorder, CD player or desktop set

Take a look at the constructor's gallery at http://www.bobrathbone.com/pi_radio_constructors.htm to get some ideas that other constructors have used.

Note: Don't forget to make sure that there is adequate airflow through the radio housing to allow cooling of the Raspberry PI and other components. Drill at least five or six holes at the top and bottom of the housing.

Wiring

The following table shows the GPIO LCD interface wiring for both the push button and rotary encoder versions of the radio. If using the Adafruit LCD plate, skip this section (See *Construction using an Adafruit* LCD plate on page 21). This wiring works for both revisions 1 and 2 boards of the Raspberry PI however revision 2 boards require a small code change in the *lcd_class.py* and radio daemon code (radiod.py etc). Refer to RPI Low level Peripherals page at http://elinux.org/RPi_Hub for more information on Raspberry PI low level peripheral wiring.

Table 2 Controls and LCD wiring

Pin	Description	Function	LCD pin	Push Buttons	Encoder (Tuner)	Encoder (Volume)
1	3V3	Push button supply		COMMON		
2	5V	5V for LCD	2,15			
3	GPIO 0					
4	Reserved					
5	GPIO1					
6	GND	Zero volts	1,3*,5,16		Common	Common
7	GPIO 4	Mute volume				Knob Switch
8	GPIO 14	Volume down		LEFT		Output A
9	Reserved					
10	GPIO 15	Volume up		RIGHT		Output B
11	GPIO 17	Channel Up		UP	Output B	
12	GPIO 18	Channel Down		DOWN	Output A	
13	GPIO 27(21)*	LCD Data 4	11			
14	Reserved					
15	GPIO 22	LCD Data 5	12			
16	GPIO 23	LCD Data 6	13			
17	Reserved					
18	GPIO 24	LCD Data 7	14			
19	GPIO 10					
20	Reserved					
21	GPIO 9					
22	GPIO 25	Menu Switch		MENU	Knob Switch	
23	GPIO 11					
24	GPIO 8	LCD E	6			
25	Reserved					
26	GPIO 7	LCD RS	4			

* LCD Pin 3 (Contrast) may be connected to the centre tap of a 10K preset potentiometer. See page 15. Pin 13 is GPIO27 on Rev 2 boards and GPIO21 on Rev 1 boards

Note: Make sure you are using the correct columns in the above table. Use column 5 (Push Buttons) for the push button version and the last two columns (Encoder Tuner/Volume) for the rotary encoder version.

Version 2 boards (latest)

Wire one side of the switches to the 3.3V pin. Wire the other side of each switch to the GPIO pin as shown in the last column of the above table. This is the normal option. Version 2 boards have internal pull up/down resistors and don't require external resistors.

Version 1 boards (early boards)

Wire one side of the switches to the 3.3V pin. Wire the other side of each switch to the GPIO pin as shown in the last column of the above table via a 1KΩ resistor. Also wire this same side of the switch to the 0V pin via a 10KΩ resistor. See Figure 11 on page 14.

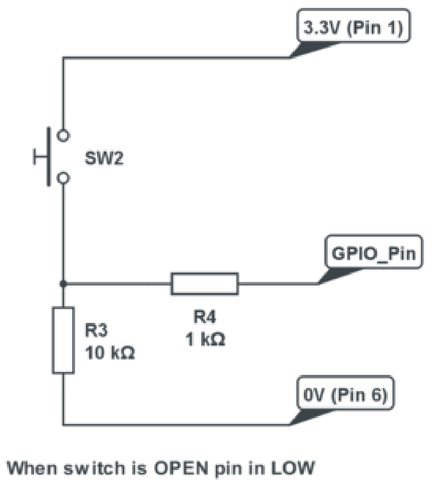


Figure 11 Switch Wiring version 1 boards

Note: The circuit will work without the 1KΩ resistor but is advised for extra protection for the GPIO input.

Rotary encoder wiring

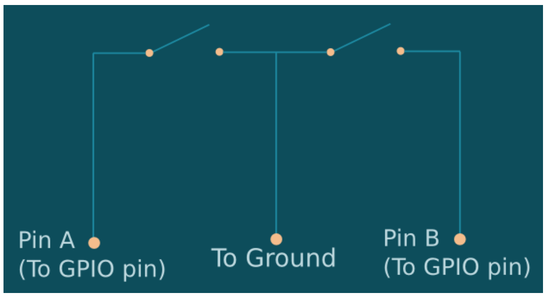


Figure 12 Rotary Encoder Diagram

Rotary encoders have three inputs namely Ground, Pin A and B as shown in the diagram on the left. Wire the encoders according shown in Table 2 on page 13. If the encoder also has a push button knob then wire one side to ground and the other to the GPIO pin. In the case of the mute switch this will be pin 7 (GPIO 4).

If using a Rev 1 board it is necessary to use 10K pull up resistors connected between the GPIO inputs and the 3.3 volt line.

This project uses a COM-09117 12-step rotary encoder from Sparkfun.com. It also has a select switch (by pushing in on the knob).

Figure 13 Rotary encoder with push switch

LCD Module Wiring

The following shows the wiring for the HD44780 LCD controller. It has 16 or 18 pins.

1. Ground (0V)
2. VCC (Usually +5V)
3. Contrast adjustment (0V gives maximum contrast)
4. Register Select (RS).
   RS=0: Command, RS=1: Data
5. Read/Write (R/W). Very important this pin must be grounded!
   R/W=0 (GND): Write, R/W=1 (+5V): Read. Will damage the PI if not grounded.
6. Enable
7. Data Bit 0 (Not required in 4-bit operation)
8. Data Bit 1 (Not required in 4-bit operation)
9. Data Bit 2 (Not required in 4-bit operation)
10. Data Bit 3 (Not required in 4-bit operation)
11. Data Bit 4
12. Data Bit 5
13. Data Bit 6
14. Data Bit 7
15. LED Backlight Anode (+5V) or Red LED (Adafruit RGB plate)
16. LED Backlight Cathode (GND)
17. Optional Green LED (Adafruit RGB plate)
18. Optional Blue LED (Adafruit RGB plate)

If the LCD being used comes with a 10K preset potentiometer supplied, then connect pin 3 of the LCD to the centre pin of the potentiometer. Connect the other two pins of the potentiometer to 5v and 0v respectively. Adjust the preset potentiometer for the best contrast.

The standard LCD comes with 16 pins. Adafruit supply an RGB backlit LCD with two extra pins namely pins 17 and 18. These are non-standard. These must be wired to ground (0 Volts) to work. For more information see the section *Using the Adafruit backlit RGB LCD display* on page 59.

Usually the device requires 8 data lines to provide data to Bits 0-7. However the device can be set to a “4 bit” mode which allows data to be sent in two chunks (or nibbles) of 4 bits. This is advantageous as it reduces the number of GPIO connections required by four when interfacing with the Raspberry Pi.

Power supply considerations

The Raspberry Pi uses a standard Micro USB (type B) power connector, which runs at 5 V. In general the Raspberry PI can draw up to 700mA. Many telephone adapters deliver less than that and can lead to problems. You also need to consider the LCD screen which can also need up to 20mA but depends on the type of backlight.

Try to find an adapter that delivers at least 1.0 Ampere. Even better is 1.5 Amperes. See the following article.

http://elinux.org/RPi_VerifiedPeripherals#Power_adapters

The Raspberry PI can be powered either the USB port or via the GPIO header (Pin 2). Some prototyping boards such as the Ciseco Humble PI can provide power in this way.
See http://shop.ciseco.co.uk/k001-humble-pi/
This interface board can be ordered with an optional 5 volt regulator. If using the Humble PI try with regulator use 6v to 7v as the power input to the Humble PI 5v regulator (Not the Raspberry PI). Trying to use 9v or more will mean that the 5 volt regulator will get far too hot.

If using an adaptor or separate 5 volt Power Supply try to use a switched-mode power supply adaptor. This take less current and generate less heat that a power dissipation device.

Things not to do:

- Do not try to tap off power from the Power supply or transformer used by the speaker's amplifier. This won't work (earth loops) and can cause damage to the PI and peripherals.
- Do not feed power to the PI from two sources (USB hub and Power adapter). Try to use USB hubs that don't feed 5 volts back through the USB ports of the Raspberry PI
- Do not connect an untested power supply to the Raspberry PI without checking the voltage first.
- Do not try to power the USB hub from the Raspberry PI. It is unlikely to cope. Use a separately powered USB hub.

You should try to use a single power supply switch for the radio. One technique which is quite useful is to split open a USB power adaptor (Use a hacksaw very carefully). Connect the AC power supply of the adaptor to the mains switch. This switch can also provide the mains supply to the speaker amplifier.

Always consider safety first and make sure that no-one including yourself can receive an electric shock from your project including when the case is open.

GPIO Hardware Notes

The following shows the pin outs for the GPIO pins. For more information see:
http://elinux.org/RPi_Low-level_peripherals

Raspberry PI GPIO numbering

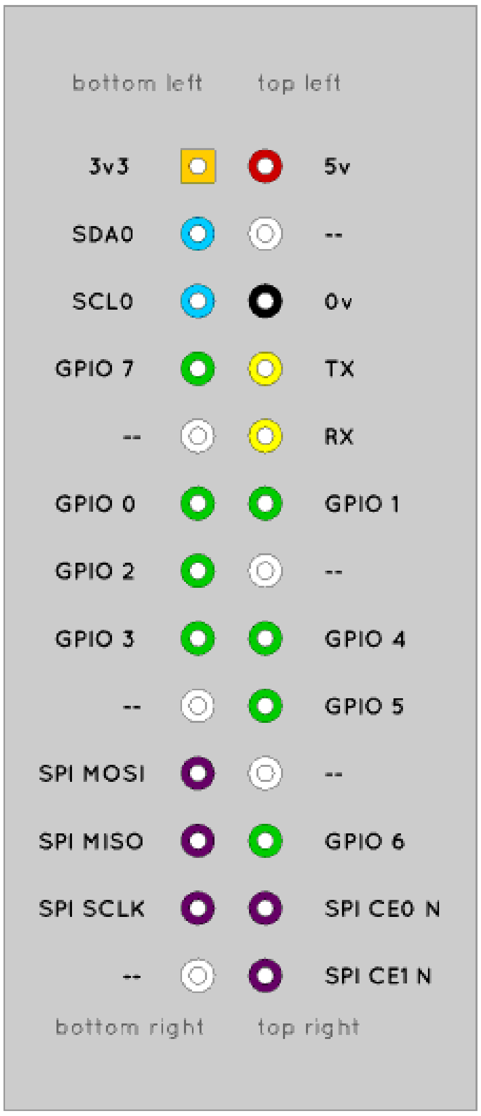
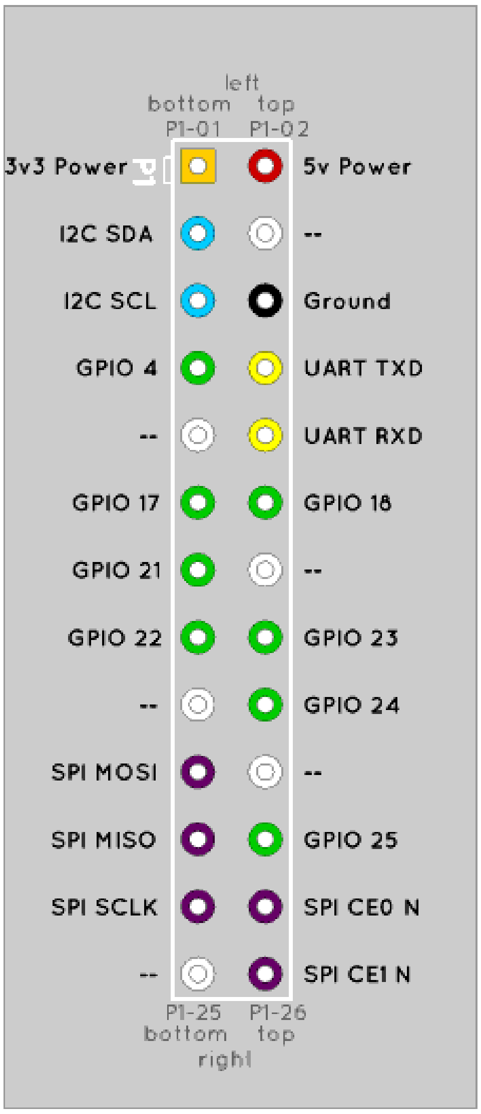


Figure 14 GPIO wiring

Raspberry GPIO Broadcom numbering



Note: On rev 2 boards GPIO21 is now GPIO27

Parts List

The following table shows the parts list for the Raspberry PI Internet Radio. This list is for the version using the HD44780 LCD directly connected to the GPIO pins. If using the Adafruit five button LCD Plate then don't order the parts marked with an asterix (*)

Table 3 Parts list

Qty	Part	Supplier
1	Raspberry Pi Computer	Farnell Element 14
1	Clear Raspberry Case	RS Components
1	4GByte SD Card	Any PC or Photographic supplier
1	Wooden Radio Case	A good friend of mine
1	Raspbian Wheezy OS	Raspberry Pi foundation downloads
2	Four inch loudspeakers	From set of old PC speakers
2	Four inch loudspeaker grills	Any electronics shop
1	Stereo Amplifier (3 to 5 watt)	From set of old PC speakers
1	Transformer for amplifier	From set of old PC speakers
1	LCD HD44780 2 x 16 Display *	Farnell Element 14
1	ModMyPi Slice of Pi *	Ciseco PLC
4	Round push buttons *	Any electronics shop
1	Square push button *	Any electronics shop
2	Rotary encoders if using this option * **	Sparkfun.com
1	26 way ribbon cable	Tandy or Farnell Element 14
5	10KΩ resistors * (Version 1 boards only)	Tandy or Farnell Element 14
5	1K resistors * (Version 1 boards only)	Tandy or Farnell Element 14
1	Four port USB hub	Any PC supplier
1	External power supply for USB hub (1200 mA)	Any PC supplier
1	26 way PCB mount male connector	Any electronics shop
1	Mains cable	Hardware shop
1	Double pole mains switch with neon	Farnell Element 14
5	Male 2 pin PCB mount connectors	Any electronics shop
2	Female 4 pin PCB connectors	Any electronics shop
1	Female 2 pin PCB connectors	Any electronics shop
1	16 pin male in-line PCB mount connector	Any electronics shop
1	Stereo jack plug socket	Any electronics shop
1	Wall mount Ethernet socket	Any do-it-self shop
	Shrink wrap	Any electronics shop
	Thin wire for PCB wiring	Any electronics shop

* These components are not required if using the Adafruit LCD plate.
** If using rotary encoders.

Construction HD44780 LCD

The following is for an HD44780 LCD display directly wired to the GPIO pins. If using the Adafruit LCD plate see section called *Construction using an Adafruit* LCD plate on page 21.

The following illustration shows the parts for the classic style radio.

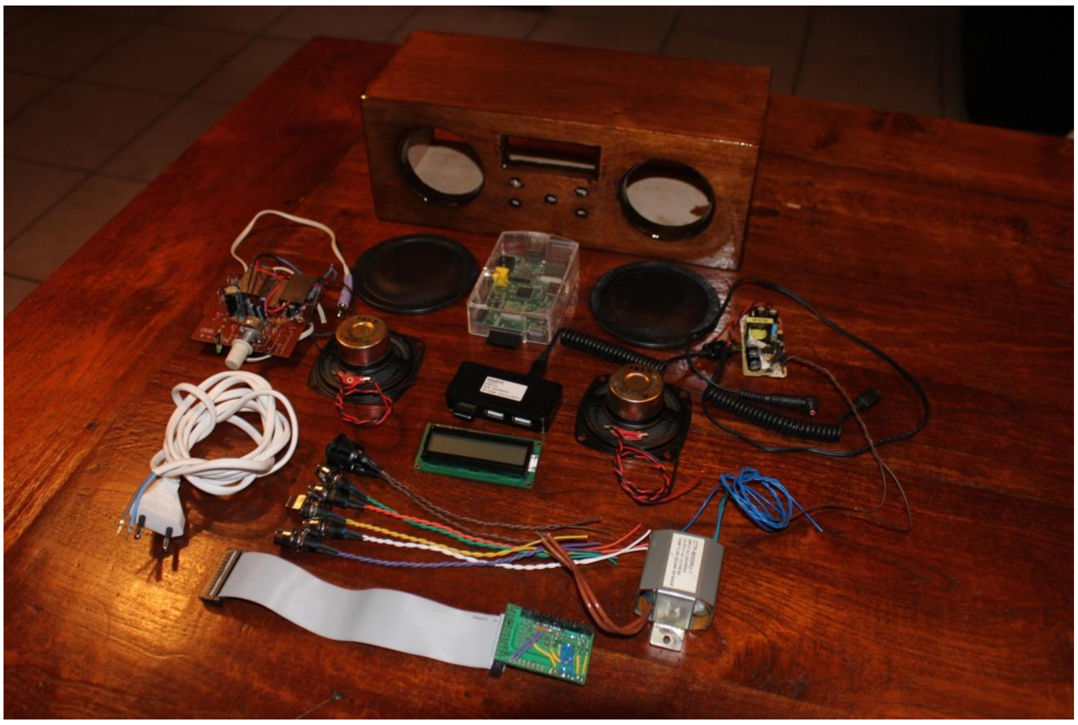


Figure 15 Radio parts

The above photo shows the components before assembly. See from back and left to right. A wooden case, mains cable, Raspberry PI in a transparent plastic case, speaker grills, 5v power supply, 4 inch speakers, 2 x 16 LCD display, four port USB hub, stereo amplifier, five push button switches, 11.5v transformer for the stereo amplifier, ribbon cable and the LCD and push buttons interface board. Not shown is the (optional) USB wireless dongle.

Building the LCD and pushbuttons interface board

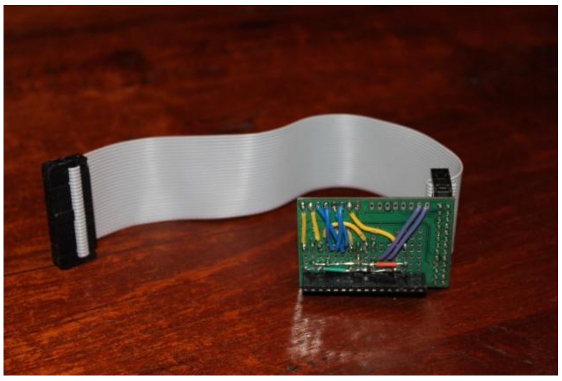


Figure 16 Interface board (Wiring side)

The LCD and push button interface was built using a ModMyPi Slice of PI from Ciseco PLC but a suitable prototype board will do. The board was fitted with 1 female 16 pin in-line connector for LCD and a male 26 pin connector for the 26 way ribbon cable. If no ribbon cable is to be used then use a female 26 way connector to plug into the Raspberry PI GPIO interface

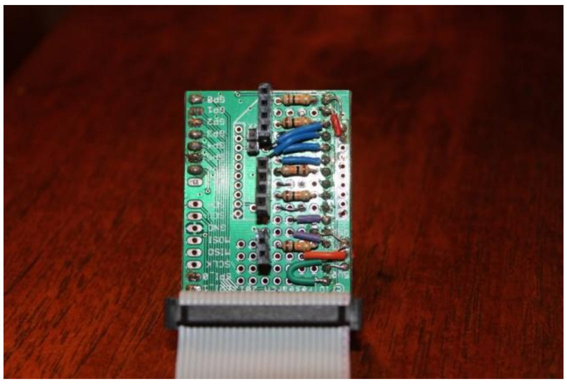


Figure 17 Interface board (Component side)

The component side of the LCD and push button shows the female connectors for five push button switches and the five 10KΩ pull down resistors. In this construction the 1KΩ resistors shown in Figure 11 Switch Wiring on page 11 weren't used but do provide some extra protection for GPIO inputs.

The following picture shows the components mounted in the wooden case.

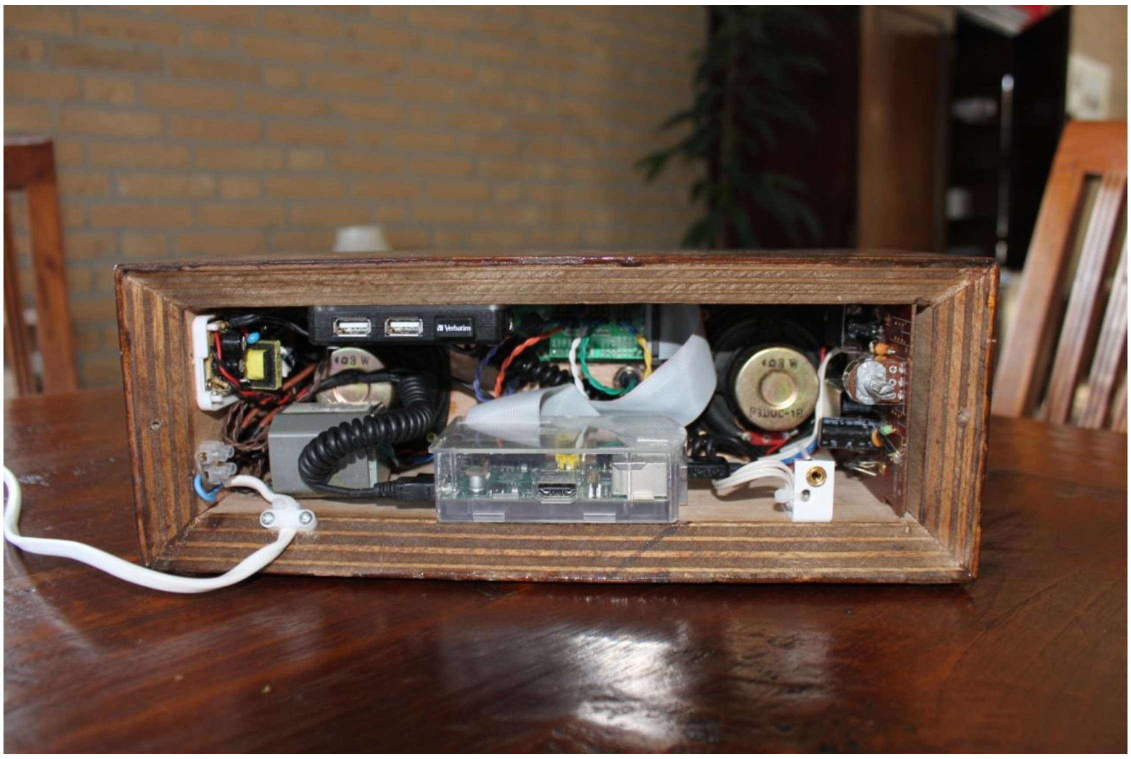


Figure 18 Radio rear inside view

Shown from top left to bottom right: 5V power supply (from a standard phone charger), four port USB with a memory stick for music files, LCD and Switch interface board (You can just see one of the switches connected with a twisted green wire), stereo amplifier (volume control now just a preset) and 4 inch speakers from a set of old PC speakers, mains input (mains switch behind this), mains transformer for the amplifier, Raspberry PI in a clear plastic case and finally the headphones socket.

Construction using an Adafruit LCD plate

This section is for the radio using an Adafruit RGB-backlit LCD plate for Raspberry PI. The complete instructions for both ordering and building this product are available from the following web site.

http://www.adafruit.com/products/1110 (See tutorials)

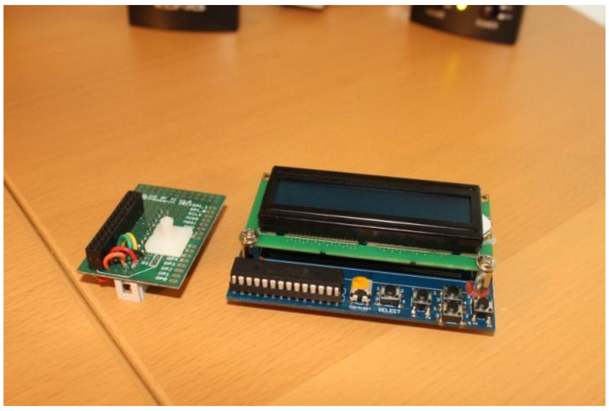


Figure 19 Adafruit LCD plate

The Adafruit LCD plate is designed to directly into the GPIO header on the Raspberry PI. These fit into a female 26 way header. If you want to connect the Adafruit LCD via a ribbon cable you will need to mount a 26 way male header instead of the female header and you will also need to construct a reversing board (Shown on the left of the picture on the left). Because ribbon cable swaps the two rows of pins over the reversing card is required to swap the two rows of pins back to their correct orientation

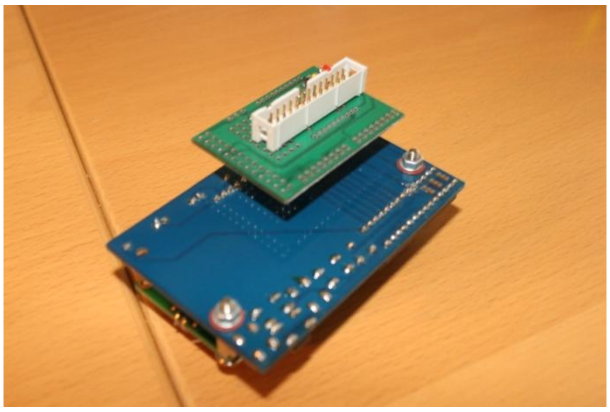


Figure 20 Adafruit LCD plate with ribbon cable adapter

Back view of the reversing board plugged into the Adafruit LCD plate.

The GPIO pins used are

1. 3.3 Volts
2. 5.0 volts
3. SDA0
4. -
5. SCL0
6. Ground

Note 1: If you are going to plug the Adafruit LCD plate directly into the GPIO header on the Raspberry PI then you don't need the above reversing plate. Just follow the construction instructions on the tutorial on the Adafruit site.

Note 2: The select button on the Adafruit plate is the "Menu" button for the radio.

Note 3: If you want to use an Adafruit display that allows setting different colours for the backlight then see section *Configuring the Adafruit LCD backlight colours* on page 58 for instructions on how to do this.

Software installation

SD card creation

Create an SD card running the latest version of Raspbian Wheezy. This can be downloaded from http://www.raspberrypi.org/downloads. See the *Image Installation Guides* on this page for instructions on how to install the Raspian Wheezy operating system software.

Conventions used in this tutorial

Installation of the radio program requires you to enter lines at the command line prompt. This requires you to log into the Raspberry PI as user 'pi'. The default password is raspberry.

```
Raspberrypi login: pi
Password: raspberry
pi@raspberrypi:~$ Last login: Sun Apr  6 10:18:18 2014 from 192.168.2.100
pi@raspberrypi:~$
```

The prompt line is displayed ending with a **$** sign. The **pi@raspberrypi:~** string means user 'pi' on host machine called 'raspberrypi'. The **~** character means the user 'pi' home directory. In this tutorial if you are required to do something as user pi then only the $ sign will be shown followed by the command as shown in the example below:

```
$ mpc status
```

Some of the commands need to be carried out as user 'root'. To become root user type in the 'sudo bash' command:

```
$ sudo bash
root@raspberrypi:/home/pi#
```

Again the prompt shows the username, hostname and current working directory. However only the # followed by the required command will be shown in this tutorial. For example:

```
# apt-get install mpd mpc
```

Online update and upgrade of the Operating System

Once you have installed the operating system login to the system and run the following commands as root user:

```
# apt-get update
```

If you don't carry out this step then installation of the Music player daemon may fail.

Radio Software Installation

This procedure assumes that the Raspberry PI is installed with Debian Wheezy and with a working Internet Connection. There are five steps to carry out to install the software.

- Download and un-tar the software from the Bob Rathbone web site
- Install the I2C library (Adafruit plate version only)
- Install the Music Player Daemon (MPD) and Client (MPC)
- Create play lists for the MPD and test

- Configure and run the radio software
- Optionally install Icecast2 streaming software

Most instructions will have to be carried out as user root. To do this log in as user pi and run *sudo bash*.

Software download

The software is contained in a compressed tar file called *pi_radio.tar.gz*. This can be downloaded from the following location.

http://www.bobrathbone.com/pi_radio_source.htm

Either download it to the PC and copy it across to the Raspberry PI or use the *wget* facility if there is an Internet connection on the Raspberry PI.

Create a directory called **/home/pi/radio**. Copy the *pi_radio.tar.gz* to the **/home/pi/radio** directory or use **wget** to download it.

```
# cd /home/pi/radio
# wget http://www.bobrathbone.com/raspberrypi/source/pi_radio.tar.gz
```

Note if you carry out this command a second time it will rename the file. Remove the old pi_radio.tar.gz file first.

Un-tar the file with the following command:

```
# tar -xvf pi_radio.tar.gz
```

This will unzip all of the required programs and configuration files into **/home/pi/radio.** Please refer to the **README** file for important release information.

Make sure all of the *.py files in this directory are executable with the following command:
# chmod +x *.py.

Don't try to run the software just yet! It has to be configured first.

Install the I2C libraries

If you are using the Adafruit plate it is necessary to install the I2C libraries. If using the LCD directly wired to the GPIO pins then skip this section.

For a more basic introduction to setting up I2C on your Pi then you may wish to take a look at this Adafruit tutorial: http://learn.adafruit.com/adafruits-raspberry-pi-lesson-4-gpio-setup/configuring-i2c

Edit **/etc/modules** file as sudo or user root and add the following lines to the end of the file. Then save and reboot to enable the hardware I2C driver.

```
i2c-bcm2708
i2c-dev
```

Reboot the system:

```
$ sudo reboot
```

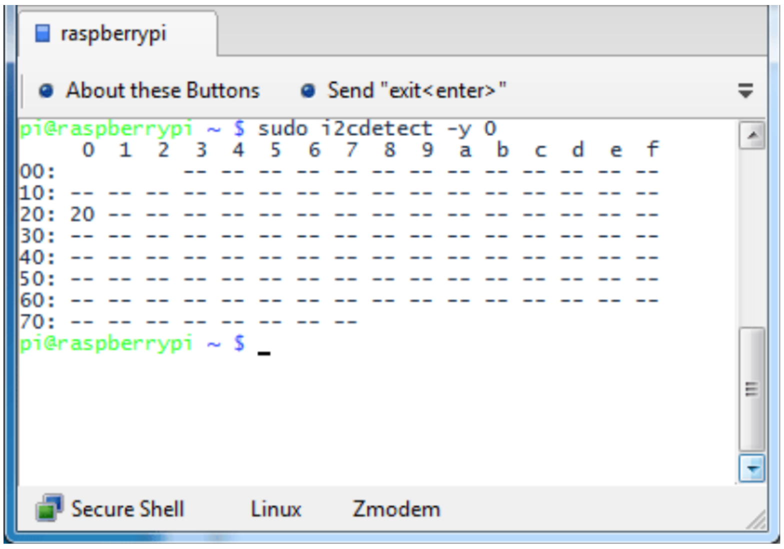
After rebooting log back in and enter the following commands to add SMBus support (which includes I2C) to Python:

```
$ sudo apt-get install python-smbus
$ sudo apt-get install i2c-tools
```

The **i2c-tools** isn't strictly required, but it's a useful package since you can use it to scan for any I2C or SMBus devices connected to the Raspberry. If you know something is connected, but you don't know it's 7-bit I2C address, this library has a great little tool to help you find it:

```
$ sudo i2cdetect -y 0 (if you are using a version 1 Raspberry Pi)
$ sudo i2cdetect -y 1 (if you are using a version 2 Raspberry Pi)
```

This will search /dev/i2c-0 or /dev/i2c-1 for all address, and if the Adafruit LCD Plate is correctly connected, it should show up at **0x20.**



```
pi@raspberrypi ~ $ sudo i2cdetect -y 0
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
20: 20 -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
pi@raspberrypi ~ $ _
```

Figure 21 The I2C bus display using the i2cdetect program

Once both of these packages have been installed, you have everything you need to get started accessing I2C and SMBus devices in Python.

Checking Network Time Daemon

Since the radio displays the time it is necessary to sync the time with a Network Time Protocol (NTP) server. In the latest versions of Raspian Wheeze OS the NTP daemon is started automatically.

Test that the time is synchronising OK with the **ntpq peers** command

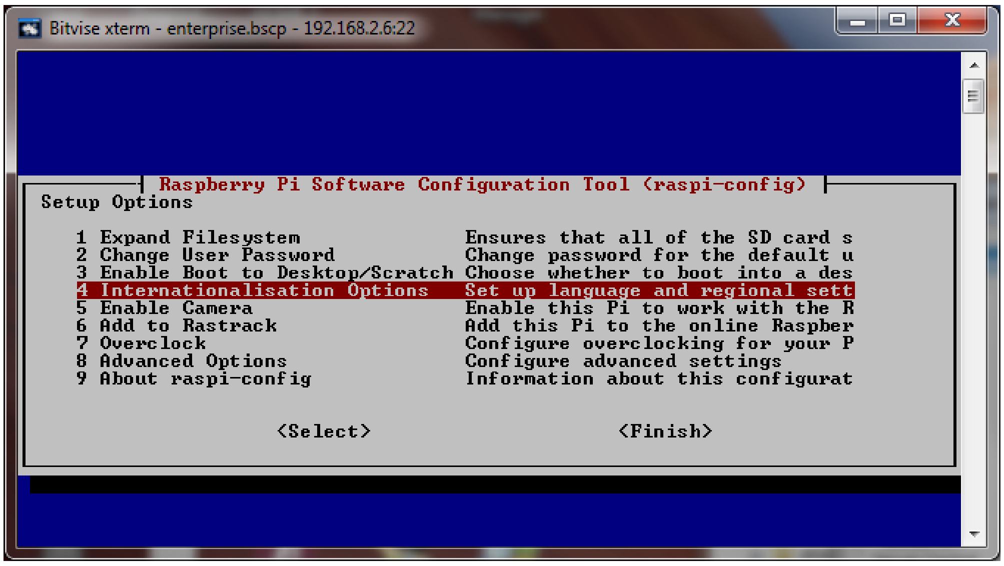
```
$ sudo ntpq
ntpq> peers
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
+ns0.solcon.nl   193.79.237.14    2 u  438  512  377   20.337   -1.030   5.302
-mu.monshouwer.e 193.79.237.14    2 u  352  512  377   20.197   -1.858   0.349
*ev001.tilaa.nl  193.190.230.65   2 u  415  512  377   19.891   -0.899   0.379
+ntp.raqxs.nl    212.45.32.36     3 u  388  512  377   20.354   -1.127   0.391
ntpq> quit
```

Note: Other servers will be displayed depending upon your time zone.
If NTP isn’t running correctly refer to resources on the Internet to get it running.

Setting the time zone

The Debian Squeeze operating system is usually set to UK time. The easiest way to set the time zone for your country if you are in a different timezone is to use the **raspi-config** program.

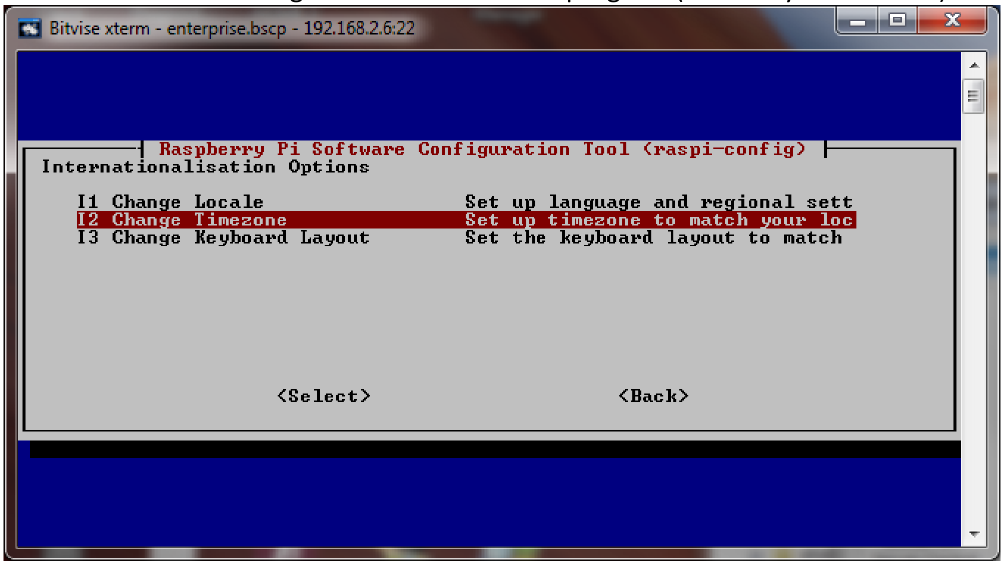
```
$  sudo raspi-config
```



```
Raspberry Pi Software Configuration Tool (raspi-config)
Setup Options

   1 Expand Filesystem              Ensures that all of the SD card s
   2 Change User Password           Change password for the default u
   3 Enable Boot to Desktop/Scratch Choose whether to boot into a des
   4 Internationalisation Options   Set up language and regional sett
   5 Enable Camera                  Enable this Pi to work with the R
   6 Add to Rastrack                Add this Pi to the online Raspber
   7 Overclock                      Configure overclocking for your P
   8 Advanced Options               Configure advanced settings
   9 About raspi-config             Information about this configurat

                    <Select>                     <Finish>
```

Figure 22 raspi-config start screen

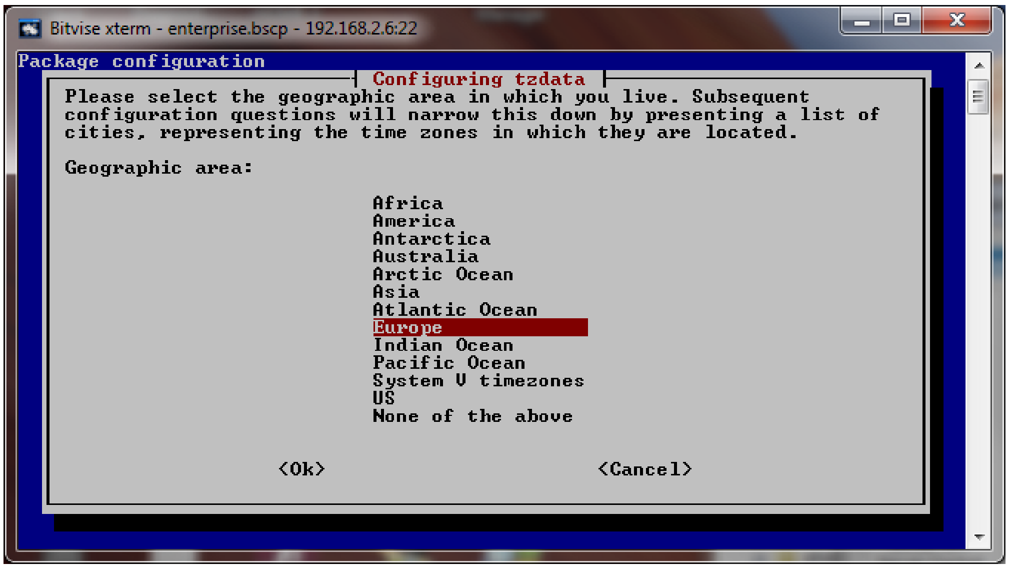
Select option 4 “Internationalisation Options”, Use the tab key to move to <Select> and press enter. The above screen is using the Bitvise SSH client program (See Putty on the web).



```
Raspberry Pi Software Configuration Tool (raspi-config)
Internationalisation Options

   I1 Change Locale                 Set up language and regional sett
   I2 Change Timezone               Set up timezone to match your loc
   I3 Change Keyboard Layout        Set the keyboard layout to match

                    <Select>                     <Back>
```

Figure 23 Selecting the time zone

Select the “Change Timezone” option. Again use the tab key to move to <Select> and press enter.



```
Bitvise xterm - enterprise.bscp - 192.168.2.6:22
Package configuration
                         ┤ Configuring tzdata ├
  Please select the geographic area in which you live. Subsequent
  configuration questions will narrow this down by presenting a list of
  cities, representing the time zones in which they are located.

  Geographic area:

                          Africa
                          America
                          Antarctica
                          Australia
                          Arctic Ocean
                          Asia
                          Atlantic Ocean
                          Europe
                          Indian Ocean
                          Pacific Ocean
                          System V timezones
                          US
                          None of the above

                  <Ok>                        <Cancel>
```

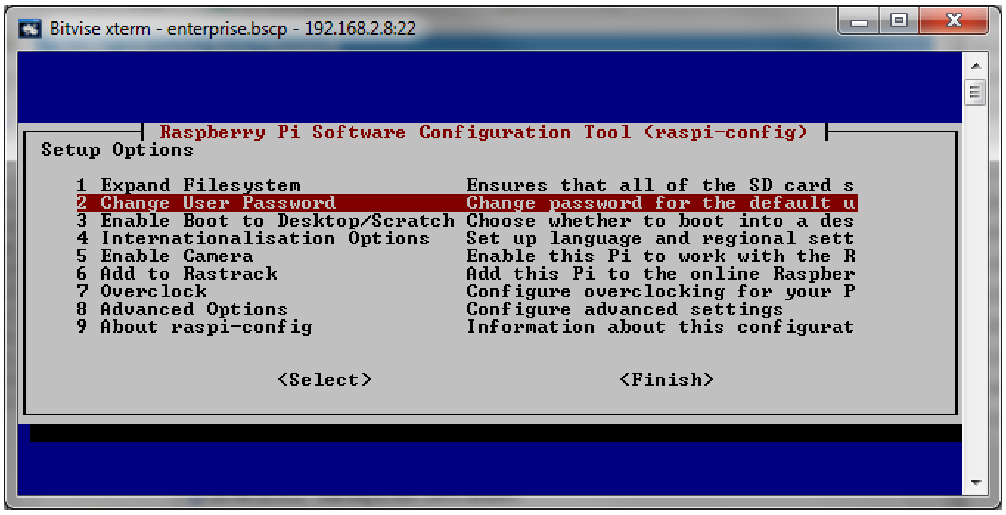
Figure 24 Saving the timezone

Select the region your country is in, Europe for example. Use the tab key to move to <OK> and press enter. The program will then display a list of timezones for the selected region. Select the correct one and save it by tabbing to <ok> and pressing the enter key. The timezone will be updated Exit the program once finished.

Changing the system hostname and password

It is a good idea to change the system password for security reasons especially if your raspberry PI is connected to a network with a wireless (WIFI) network.   Changing the hostname is also a good idea as it makes identifying your radio on the network much easier. If you wish to do this then change the default hostname from 'raspberrypi' to something like 'piradio' or 'myradio'.
Both the password and hostname can be changed using the raspi-config program.

```
$  sudo raspi-config
```

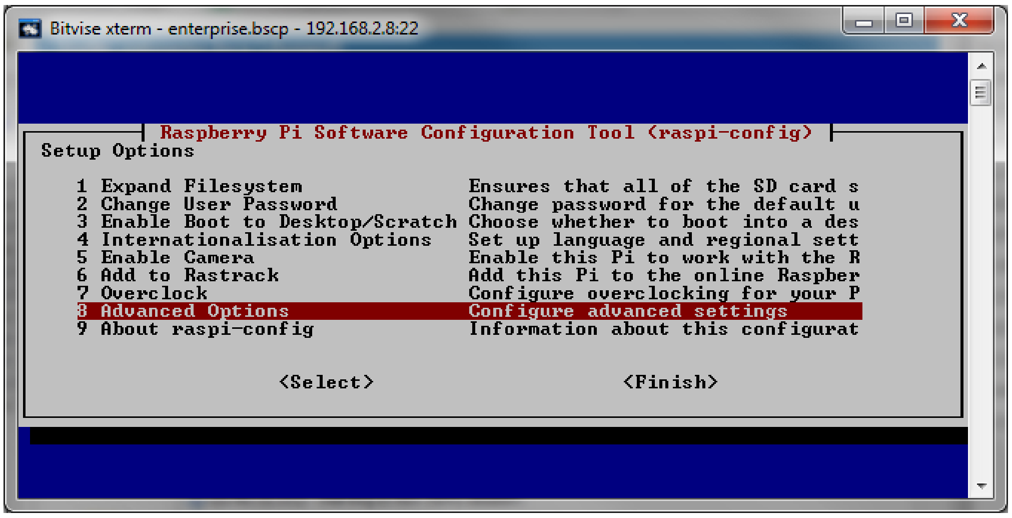


```
Bitvise xterm - enterprise.bscp - 192.168.2.8:22
          ┤ Raspberry Pi Software Configuration Tool (raspi-config) ├
Setup Options

   1 Expand Filesystem              Ensures that all of the SD card s
   2 Change User Password           Change password for the default u
   3 Enable Boot to Desktop/Scratch Choose whether to boot into a des
   4 Internationalisation Options   Set up language and regional sett
   5 Enable Camera                  Enable this Pi to work with the R
   6 Add to Rastrack                Add this Pi to the online Raspber
   7 Overclock                      Configure overclocking for your P
   8 Advanced Options               Configure advanced settings
   9 About raspi-config             Information about this configurat

                   <Select>                     <Finish>
```

Figure 25 Changing the raspberry PI password

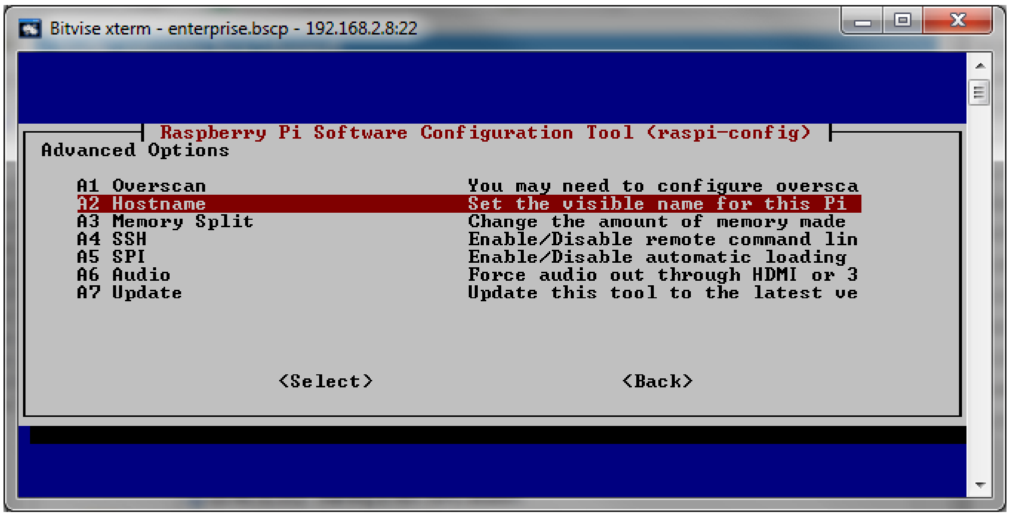
Option 2 is used to change the password. Make sure you record your new password somewhere safe (It is easy to forget it).

The hostname is changed in option 8:



```
Bitvise xterm - enterprise.bscp - 192.168.2.8:22

        Raspberry Pi Software Configuration Tool (raspi-config)
 Setup Options

    1 Expand Filesystem              Ensures that all of the SD card s
    2 Change User Password           Change password for the default u
    3 Enable Boot to Desktop/Scratch Choose whether to boot into a des
    4 Internationalisation Options   Set up language and regional sett
    5 Enable Camera                  Enable this Pi to work with the R
    6 Add to Rastrack                Add this Pi to the online Raspber
    7 Overclock                      Configure overclocking for your P
    8 Advanced Options               Configure advanced settings
    9 About raspi-config             Information about this configurat

                   <Select>                     <Finish>
```

Figure 26 raspi-config advanced options



```
Bitvise xterm - enterprise.bscp - 192.168.2.8:22

        Raspberry Pi Software Configuration Tool (raspi-config)
 Advanced Options

    A1 Overscan                      You may need to configure oversca
    A2 Hostname                      Set the visible name for this Pi
    A3 Memory Split                  Change the amount of memory made
    A4 SSH                           Enable/Disable remote command lin
    A5 SPI                           Enable/Disable automatic loading
    A6 Audio                         Force audio out through HDMI or 3
    A7 Update                        Update this tool to the latest ve

                   <Select>                     <Back>
```

Figure 27 Changing the hostname

You will be asked if you wish to reboot the system. After you reboot the system you will see the new hostname at the login prompt.

```
pi@piradio:~$
```

In the above example the new hostname is 'piradio'.

Source files

The source consists of several source modules all written in Python using Object Orientated techniques. The source can be downloaded from
http://www.bobrathbone.com/pi_radio_source.htm

The LCD Class

The LCD *lcd_class.py* class handles all of the LCD display routines. It contains simple commands to display and scroll lines of text on the HDD44780 2 x 16 LCD or 4 x 20 characters LCD. It is a useful standalone class that can be used in other projects. It is based on the routines from Matt Hawkins. If you are using a revision 1 board you will need to make a small change to the following line in the radio daemon program you are using (radiod.py etc.):

```
LCD_D4 = 21 # Rev 1 Board
#LCD_D4 = 27 # Rev 2 Board
```

If using the Adafruit LCD plate you can skip the above section.
This class can be used for other projects using the HD44780 LCD display.
For revision 1 boards only, uncomment the first five GPIO lines below and comment out the lines which enable the internal pull down resistors.

```
        # For rev 1 boards with no inbuilt pull-up/down resistors
        # Wire the GPIO inputs to ground via a 10K resistor and
        #  uncomment these lines and comment out the next block
        #GPIO.setup(MENU_SWITCH, GPIO.IN)
        #GPIO.setup(UP_SWITCH, GPIO.IN)
        #GPIO.setup(DOWN_SWITCH, GPIO.IN)
        #GPIO.setup(LEFT_SWITCH, GPIO.IN)
        #GPIO.setup(RIGHT_SWITCH, GPIO.IN)

        # For rev 2 boards with inbuilt pull-up/down resistors the
        # following lines are used instead of the above, so
        # there is no need to physically wire the 10k resistors
        GPIO.setup(MENU_SWITCH, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
        GPIO.setup(UP_SWITCH, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
        GPIO.setup(DOWN_SWITCH, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
        GPIO.setup(LEFT_SWITCH, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
        GPIO.setup(RIGHT_SWITCH,GPIO.IN, pull_up_down=GPIO.PUD_DOWN)
```

The Radio Daemon

There are four versions of this program for the HDD44780 LCD (directly wired to the GPIO pins). See

The *radiod.py* source provides the logic for operating the radio. It reads the push button switches and configuration files, loads the music files and radio stations. It is used with a 16 character two line LCD.

The *radio4.py* source provides the same logic for operating the radio but for a 20 character four line LCD.

The *rradiod.py* source is the version used with rotary encoder switches. It is used with a 16 character two line LCD.

The *rradio4.py* source is the version used with rotary encoder switches. It is used with a 20 character four line LCD.

The Adafruit Radio daemon

If you are an Adafruit RGB-backlit LCD plate for Raspberry Pi then the following programs are used:

ada_radio.py	The radio daemon for the Adafruit LCD plate.
ada_lcd_class.py	The LCD class using an I2C interface (Also interfaces the switches).
i2c_class.py	The IC2 class courtesy of Adafruit Industries (renamed).
test_ada_lcd.py	Test Adafruit LCD and switches.

The Daemon Class

The *radio_daemon.py* code allows the radio program to run as a background daemon. It allows start, stop, restart, version and status commands.

The Radio Class

The *radio_class.py* contains the actual commands that interface to the Music Player Daemon (MPD).

The Rotary class

The *rotary_class.py* configures and handles the interrupts (events) for the rotary encoders. It is used by the *rradiod.py* and *rradio4.py* programs.

The Log class

The *log_class.py* routine provides logging of events to **/var/log/radio.log** file.

The RSS class

The *rss_class.py* routines allow sequential gets from an RSS feed. These feeds are provided from news providers such as the BBC. This class gets the RSS feed defined in the **/var/lib/radiod/rss** file.

The Translate class

The *translate_class.py* is used to convert special international character sets (particularly from RSS feeds). It does this by first converting them to escape sequences and then to displayable ascii characters (These will show up in DEBUG logging). These *ascii* characters are then passed to the LCD class where they may are converted again to a valid character in the standard LCD character set.

LCD test programs

The *lcd_test.py* program provides some simple code to test the LCD. The *test_ada_lcd.py* program is used to test the LCD if you are using an Adafruit RGB-backlit LCD plate for the Raspberry PI.

Switch test programs

The *test_rotary_class.py* program can be used to check the rotary switches.

The test_switches.py program is used to test the push button wiring.

The create_playlists program

The *create_playlists.py* program is available from version 2.3 of the software onwards. It creates playlist files in the **/var/lib/mpd** directory using a list of web links (URLs) with titles as input. The original *create_playlists.py* program created a single playlist for each radio station but this wasn't particularly efficient particularly when loading the radio stations. Since version 3.5 the *create_playlists.py* program allows a group of radio stations to be grouped into a single playlist.

The operation of the *create_playlists.py* program is covered in detail in the section on managing playlist files on page 49.

The display_current program

The *display_current.py* program is available from version 3.5 of the software onwards. This is a small diagnostic program which displays the information for the current radio station or track. It is only used for trouble-shooting and it will not normally be used.

Installing and Testing the Music Player Daemon

Install the Music Player Daemon

If not already carried out do an update of the system.

```
$ sudo apt-get update
```

Install the Music Player Daemon (mpd) and its client (mpc) along with the Python MPD library.

```
$ sudo apt-get install mpd mpc python-mpd
```

Run the *create_playlists.py* program as user root or using sudo. If you are an existing user upgrading to the new software then don't do this without first copying your existing playlists to the /home/pi/radio/playlists directory. See the section Creating playlists on page 49 for more details.

```
$ cd /home/pi/radio
$ sudo ./create_playlists.py
```

This will create a set of playlist files in the **/var/lib/mpd/playlists** directory. You can display the contents of the MPD playlists directory with the *ls* command.

```
# ls /var/lib/mpd/playlists/
977_MIX_ASX.pls   Digital_Radio_108.pls  German_stations.pls
Nashville_FM.pls BBC_stations.pls  Dutch_stations.pls
Italian_stations.pls  Stream_Power_MP3.pls
```

The above list may vary from that shown depending upon the distribution list of stations downloaded with the radio software..

The operation of the *create_playlists.py* program is covered in detail in the section on managing playlist files on page 49.

Configure the MPD daemon

The following instructions should be sufficient to get the Music Player Daemon running and you should not need to otherwise alter the MPD configuration. However if you require more detail refer to the online MPD manual at: http://www.musicpd.org/doc/user/

The configuration file for the MPD daemon is **/etc/mpd.conf**. See the **mpd** man page for further information. It is only necessary to amend a couple of parameters in this file to prevent annoying start-up messages. The MPD program will run without modifying the configuration file.

Edit the **/etc/mpd.conf** file:

Change the *bind_to_address* parameter from "localhost" to "any".

```
#bind_to_address       "localhost"
```

```
bind_to_address         "any"
```

This will prevent IPv6 bind errors on start-up.

Change the *zeroconf_enabled* parameter to "no".

```
#zeroconf_enabled     "yes"
zeroconf_enabled      "no"
```

This prevents *avahi* daemon error messages on start-up.
Save the **/etc/mpd.conf** file when finished.

Test the Music Player Daemon

Connect a set of speakers or headphones to the Raspberry File and run the following commands to test the MPD daemon:

```
# service mpd start
# mpc load BBC_stations.pls
# mpc play 1
```

The BBC radio station should now be heard from Raspberry PI. If not troubleshoot the problem before continuing. To stop the MPD daemon enter:

```
# service mpd stop
```

If these tests succeed you are ready to run the Radio software.

Configuring USB speakers instead of the analogue output

It is possible to configure the radio program to use USB loudspeakers instead of the analogue output of the Raspberry PI. USB speakers such as the Logitech SB150 or Trust SP-2750p USB speakers can be used and can be powered directly from the Raspberry PI.

To enable sound over the USB speaker set edit the **/etc/modprobe.d/alsa-base.conf** configuration file. Replace the following line:

```
options snd-usb-audio index=-2
```

With the following two lines.

```
options snd-usb-audio index=0 nrpacks=1
options snd-bcm2835 index=-2
```

Save the file and reboot the Raspberry PI. The radio should now use the USB speakers (The analogue output is disabled).
Note: Make sure that the radio is first working with the analogue output before re-configuring the above file.

It may also be necessary with some USB drivers to set the **mixer_type** parameter to "software". Edit the /etc/mpd.conf file and find the following line.

```
#mixer_type                     "hardware"
```

Remove the # at the beginning of the line to enable software mixing and save the file.

```
mixer_type                      "software"
```

Restart the radio software.

Note: Unfortunately which USB drivers require this is unknown so only do this if you are having a problem with the USB speakers.

Upgrading from earlier versions

As a general rule back up your existing

From version 3.3 onwards it is necessary to install **python-mpd** library. To upgrade from earlier versions as root or sudo carry out the following command:

```
# apt-get install python-mpd
```

The ownership and permissions for **/media** and **/share** have changed. New installations automatically set these values. If upgrading from a version before version 3.4 carry out the following instructions:

Reboot the raspberry PI. Do not load the music library at this stage. Change the ownership and permissions with the following commands:

```
# chown pi:pi /media /share
```

The music library can now be loaded. There is no need to restart the radio.

If you haven't used the *create_playlists.py* program before then don't run the program straight away without copying al of your existing playlist files to the /home/pi/radio/playlists directory. Failure to do so will result in the loss all of your existing playlists. This has been changed in version 3.5 and above. The create_playlist program from version 3.5 onwards now asks if you want to delete the old playlists.

See the section *Creating playlists* on page 49.

Configure the Radio daemon

Disable serial interface

If you are using the Adafruit LCD plate – skip this section.

Two of the pins used by this design namely pin 8(GPIO8) and 10 (GPIO 15) are configured for the serial interface (UART). This must be disabled for reliable operation by removing all references to *ttyAMA0* in the **/boot/cmdline.txt** and **/etc/inittab** files

Add a hash character at the beginning of line containing ttyAMA0.
/boot/cmdline.txt

```
#dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200
console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait
```

Add the following line to the **/boot/cmdline.txt** file. This is the same line as above but without any references to ttyAMA0.

```
dwc_otg.lpm_enable=0 console=tty1 root=/dev/mmcblk0p2 rootfstype=ext4
elevator=deadline rootwait
```

In the **/etc/inittab** file disable the line which re-spawns ttAMA0 by adding a hash character at the beginning.

```
#Spawn a getty on Raspberry Pi serial line
#T0:23:respawn:/sbin/getty -L ttyAMA0 115200 vt100
```

Reboot the Raspberry PI to disable the TTY serial interface.

Configure radio program log rotation

The Radio program logs to a file called **/var/log/radio**. This can eventually fill the SD card. Create a file called **/etc/logrotate.d/radiod** with the following lines:

```
/var/log/radio.log {
      weekly
      missingok
      rotate 7
      compress
      notifempty
      copytruncate
      create 600
}
```

This will rotate the log files every week so prevent the SD card from eventually filling up. From version 3.5 onwards a file called **radiod.logrotate** is provided for this purpose. Copy this file to the **/etc/logrotate** directory as radiod.

```
# cp /home/pi/radio/radiod.logrotate /etc/logrotate.d/radiod
```

Configure the Radio program to start automatically at boot time

As from version 3.1 onwards a so called *System V startup* script called **radiod** has been added to start the radio daemon automatically.

Earlier versions required adding the radio daemon command to the **/etc/rc.local** file and if you are upgrading to version 3.1 then this approach will still work.

For fresh installations of the software copy the **/home/pi/radio/radiod** script to the **/etc/init.d** directory:

```
# cp radiod /etc/init.d/.
```

Now edit the **/etc/init.d/radiod** file and find the line called **NAME=radio4.py**.

```
# Change NAME parameter this next line to the version of the daemon you are
using
# Choices are radiod.py, radio4.py, rradiod.py, rradio4.py or ada_radio.py
# No spaces around the = character
NAME=radio4.py
```

You must now change the name of the program you will be using as shown in Table 1 on page 12. For example if you are using the Adafruit LCD plate then NAME becomes:

```
NAME=ada_radio.py
```

Make sure there are no spaces around in the above line.

Now enable the **radiod** startup script with the **update-rc.d** command

```
# update-rc.d radiod defaults
update-rc.d: using dependency based boot sequencing
```

Ignore warnings about insserv.

This now allows starting and stopping of the radio daemon with the **service** command:

```
# service radiod start
# service radiod stop
```

Reboot the Raspberry PI. The radiod program should start at boot time.
To prevent automatic start-up at boot time run the following command:

```
# update-rc.d radiod disable
```

To re-enable it:

```
# update-rc.d radiod enable
```

Operation

This section assumes that the LCD screen is working correctly, the MPD daemon is installed and tested and that there is an Internet connection available. If you are using a 4x20 LCD then substitute *radiod.py* for *radio4.py* in all of the following commands. If using the Adafruit LCD plate then substitute *ada_radio.py* for all of the following commands. If using rotary switches use *rradiod.py* and *rradiod.py* instead.

Starting the program

The program must either be run as root user or using sudo.
The basic operation of the program is:

```
radiod.py start|stop|restart|status|version
```

Where start: Start the radio program.
stop: Stop the radio program.
restart: Restart the radio program.
status: Show the status of the radio daemon.
version: Show the version number of the program

Change to the **/home/pi/radio directory** and run the following command:

```
pi@raspberrypi:~$ cd /home/pi/radio/
pi@raspberrypi:~/radio$ sudo ./radiod.py start
```

Alternatively run as root. All commands as root user don't need sudo.

```
# pi@raspberrypi:~$ sudo bash
# cd /home/pi/radio
# ./radiod.py start
```

To stop the radio

```
# ./radiod.py stop
```

For the 4 x 20 character LED run:

```
# ./radio4.py start
```

To display the status run:

```
# ./radiod.py status
radiod running pid 2098
```

The above pid (Process ID) number will be different each time the program is run.
To see what version of the software you are running:

```
# ./radiod.py version
Version 3.5
```

If the **radiod** service script has been configured as shown in the section *Configure the Radio program to start automatically at boot time* on page 36 then the service command may also be used:

```
# service radiod start
```

```
# service radiod stop
Stopping Music Player Daemon: mpd.
```

The **status** and **version** commands will also work:

```
# service radiod status
radiod running pid 1957
```

```
# service radiod version
Version 3.5
```

The advantage of using these commands is that they are the same for all versions of the radio as this is configured in the **radiod** service start/stop script.

Buttons

There are five buttons, four function buttons and one menu button. The Menu button changes the display mode and the functions of the left and right hand buttons as shown in the following table. If using rotary encoders please see Table 5 on page 40.

Table 4 Push Button Operation

	Left hand buttons		**Right hand buttons**	
LCD Display Mode	**Left button**	**Right button**	**Left button**	**Right button**
Mode = TIME Line 1: Time Line 2: Station or Track	Volume Up	Volume Down	Station/Track up	Station/Track down
Mode = SEARCH If source = RADIO Line 1: Search: Line2: Radio Station	Volume Up	Volume Down	Scroll up radio station	Scroll down radio station
Mode = SEARCH If source = MUSIC LIBRARY Line 1: Search Line2: MusicTrack/Artist	Scroll up through artists	Scroll down through artists	Scroll up through track	Scroll down through track
Mode = SOURCE Line 1: Input Source: Line2: Internet Radio or Music Library	Volume Up Mute	Volume Down Mute	Toggle mode between Radio and Music Library	Toggle mode between Radio and Music Library
Mode = OPTIONS Line 1: Menu Selection Line 2: <option> Options are Random, Consume, Repeat, Reload Music, Timer, Alarm ,Alarm Time Set, Streaming:	Toggle selected mode on or off. Set timer and Alarm	Toggle selected mode on or off. Set timer and Alarm	Cycle through Random, Consume, Repeat, Reload Music, Timer, Alarm , Alarm Time Set and Streaming	Cycle through Random, Consume, Repeat, Reload Music, Timer, , Alarm Time Set and Streaming:
Mode = RSS (1) Line 1: Time Line 2: RSS feed	Volume Up Mute	Volume Down Mute	Station/Track up	Station/Track down
MODE = IP address Line 1: IP address Line 2: Station or Track	Volume Up Mute	Volume Down Mute	Scroll up through track or radio station	Scroll down through track or radio station

Note 1: If the **/var/lib/radiod/rss** file is missing or contains an invalid RSS URL then the RSS mode is skipped.

Rotary encoder operation

This option is for a radio with rotary encoders with push buttons. The volume knob when pushed in is the **Mute** sound function. Likewise the tuner knob when pushed in is the **Menu** switch.
The Menu button (Tuner knob depressed) changes the display mode and the functions of the clockwise and anti-clockwise operation of the knobs as shown in the following table.

Table 5 Rotary Encoder Knob Operation

	Volume knob		Tuner knob	
LCD Display Mode	**Clockwise**	**Anti-clockwise**	**Clockwise**	**Anti-clockwise**
Mode = TIME Line 1: Time Line 2: Station or Track	Volume Up	Volume Down	Station/Track up	Station/Track down
Mode = SEARCH If source = RADIO Line 1: Search: Line2: Radio Station	Volume Up	Volume Down	Scroll up radio station	Scroll down radio station
Mode = SEARCH If source = MUSIC LIBRARY Line 1: Search Line2: MusicTrack/Artist	Scroll up through artists	Scroll down through artists	Scroll up through track	Scroll down through track
Mode = SOURCE Line 1: Input Source: Line2: Internet Radio or Music Library	Volume Up Mute	Volume Down Mute	Toggle mode between Radio and Music Library	Toggle mode between Radio and Music Library
Mode = OPTIONS Line 1: Menu Selection Line 2: <option> Options are Random, Consume, Repeat, Reload Music, Timer, Alarm and Alarm Time set.	Toggle selected mode on or off. Set timer and Alarm	Toggle selected mode on or off. Set timer and Alarm	Cycle through Random, Consume, Repeat, Reload Music, Timer, Alarm Time Set and Streaming	Cycle through Random, Consume, Repeat, Reload Music, Timer, Alarm , Alarm Time Set and Streaming
Mode = RSS (1) Line 1: Time Line 2: RSS feed	Volume Up	Volume Down	Station/Track up	Station/Track down
MODE = IP address Line 1: IP address Line 2: Station or Track	Volume Up	Volume Down	Scroll up through track or radio station	Scroll down through track or radio station

Note 1: If the **/var/lib/radiod/rss** file is missing or contains an invalid RSS URL then the RSS mode is skipped.

Mute function

Pressing both volume buttons together or in the case of a rotary encoder with a push button (Volume) will mute the radio. Press either the volume up or down switch to un-mute the radio. If you change channel or use the menu switch the radio will also be un-muted. If the alarm is set then the radio will go into sleep mode.

Playing MP3 and WMA files

The radio software also allows you to play music from the following sources:

1. From a USB stick
2. From a music partition on the SD card (Create this yourself)
3. From a Network Attached Storage (NAS)

Playing music from a USB stick

USB memory stick. Put your music tracks on a USB stick (MP3 and WMA files only) and insert it into the USB port of the Raspberry PI. Reboot the PI. Once the Radio program is running, push the Menu button until "Input source" is displayed. Press either the left or right button to change the source to "Music Library". Now press the Menu button again. The music on the USB stick will now be loaded.

Playing music from the SD card

With large (32GB) SD cards now available music can be stored on in one or more directories on the SD card. It is necessary to first create a directory in **/home/pi** as user **pi** and then link it in the /var/lib/mpd/music/ directory. Carry out the following instructions as user **pi** to create **mymusic** for example:

```
$ mkdir /home/pi/mymusic
$ cd /var/lib/mpd/music/
$ sudo ln -s /home/pi/mymusic
```

Using FTP, copy the music from a PC to the /home/pi/mymusic directory and reload the library via the options menu.

Playing music from a Network Attached Storage (NAS)

This is a bit more involved to set up. See the section called *Playing music from a Network Attached Storage (NAS)* on page 41.

Organising the music files

For the search routines to work properly the music must be organised in a certain way. The files must be placed in the top level directory of USB stick. The search routines use the first directory as the artiest name and the music files themselves as the track name. For example:

Elvis Presley/The 50 Greatest Hits Disc 1/01 That's All Right.mp3

In the above example the first directory is set up with the artiest name and this will appear in the search. The subsequent directory “The 50 Greatest Hits Disc 1” is not used by the search routines. The file name “That's All Right.mp3” without the mp3 extension becomes the track name in this example.

MPD Logging

All logging for the MPD daemon is to the **/var/log/mpd/mpd.log** file by default.

Radio program logging

The Radio program logs to a file called **/var/log/radio.log.** See example log below:

```
2014-04-06 09:42:32,185 INFO Radio running pid 12121
2014-04-06 09:42:32,211 INFO Radio ['/home/pi/radio/radio4.py', 'restart']
daemon version 3.0
2014-04-06 09:42:32,215 INFO GPIO version 0.5.4
2014-04-06 09:42:33,132 INFO Linux piradio 3.10.34+ #661 PREEMPT Thu Mar 27
00:36:02 GMT 2014 armv6l GNU/Linux
2014-04-06 09:42:33,184 INFO GPIO version 0.5.4
2014-04-06 09:42:37,050 INFO MPD started
2014-04-06 09:42:37,607 INFO mpd version: 0.16.0
2014-04-06 09:42:43,297 INFO (2) 977 MIX ASX
2014-04-06 09:42:43,510 INFO Current ID = 2
2014-04-06 10:04:52,018 INFO Radio running pid 12495
2014-04-06 10:04:52,054 INFO Radio ['/home/pi/radio/radio4.py', 'start']
daemon version 3.2
2014-04-06 10:04:52,058 INFO GPIO version 0.5.4
2014-04-06 10:04:52,976 INFO Linux piradio 3.10.34+ #661 PREEMPT Thu Mar 27
00:36:02 GMT 2014 armv6l GNU/Linux
2014-04-06 10:04:53,029 INFO GPIO version 0.5.4
2014-04-06 10:04:56,898 INFO MPD started
2014-04-06 10:04:57,458 INFO mpd version: 0.16.0
2014-04-06 10:05:03,158 INFO (2) 977 MIX ASX
2014-04-06 10:05:03,366 INFO Current ID = 2
```

There are five levels of logging namely DEBUG, INFO, WARNING, ERROR and NONE. The log level is configured in the **/var/lib/radiod/loglevel** file. The default is INFO. If you want to increase the logging say to DEBUG carry out the following command as root user (sudo won't work) and restart the program.

```
# echo DEBUG > /var/lib/radiod/loglevel
# service radiod restart
```

To switch off all logging carry out the following:

```
# echo NONE > /var/lib/radiod/loglevel
# service radiod restart
```

Configuration and status files

There are some other configuration and status files in the **/var/lib/radiod** directory. These are:

alarm	Alarm setting in t:hh:mm where t is the alarm type (t=0=off)
current_station	The current radio station
current_track	The current music track
loglevel	Log level (As previously explained)
mpdport	MPD conection port number
rss	RSS feed URL
share	The NAS share instruction
stationlist	The user list of radio station URLs
status	Stored MPD status for diagnostic purposes only
streaming	Icecast2 streamin on or off
timer	Timer (Snooze) value in minutes
volume	The volume setting

You don't normally need to change most of these files except the loglevel, stationlist,share and rss file. The others are maintained by the program so that when it starts up the program uses the last setting, for example, the volume setting.

Displaying an RSS feed

To display an RSS feed it is necessary to create the **/var/lib/radiod/rss** file with a valid RSS URL. For example:

```
http://feeds.bbci.co.uk/news/uk/rss.xml?edition=int
```

The above is the RSS for the BBC news however any valid RSS feed may be used. If the **/var/lib/radiod/rss** is missing or contains an invalid RSS URL then this mode is skipped when stepping through the menu. The software is provided with a valid BBC RSS feed file in the **rss** directory. You can test the feed first by pasting it into your PC's web browser URL and pressing enter.

Using the Timer and Alarm functions

Since version 3.0 of the radio program there is a timer (Snooze) and alarm function. The timer and alarm can operate individually or together. The timer when set will put the radio into Sleep Mode when the timer expires. The Alarm can be set to either On, Repeat or "Weekdays only".

Setting the Timer (Snooze)

Press the Menu button until the "Menu Selection" is displayed. Press either the channel UP or DOWN control until "Timer off" is displayed on line 2 of the LCD screen. Now push the volume UP button to set the timer. Use volume UP and DOWN to adjust the timer which will be displayed as "Timer hh:mm:ss" wher hh=hours, mm=minutes and ss=seconds. The Timer can be set up to 24 hours in increments of one minute. Once the timer is set, press the Menu button; the display will return to TIME mode.

On a four line LCD display the timer will be seen counting down after the Volume display on line 4. On a two line LCD display the timer count down will be displayed on line 1 after the time display.

When the timer expires (reaches zero) the radio will enter SLEEP mode. Sleep mode can only be exited by pressing the menu button.

To switch the timer off go back to the timer menu as described above and reduce the timer to 0 using the volume DOWN control. This will switch off the timer.

The timer function uses the **/var/lib/radiod/timer** file which will contain the value of the timer in minutes when it was successfully fired. You do not need to change the contents of this file.

Setting the Alarm

The Alarm menu has two settings:

- The alarm type (On, off, repeat etc)
- The Alarm time (Pressing menu in this mode puts the radio into Sleep mode)

Press the Menu button until the "Menu Selection" is displayed. Press either the channel UP or DOWN (Or rotate rotary switch) until "Alarm off" is displayed on line 2 of the LCD screen. Using the volume UP control cycle through the options which are

- Alarm off - The Alarm is switched off
- Alarm on – The Alarm is on for one time only. Once the alarm is fired it will return to off.
- Alarm repeat – The Alarm will be repeated every day and not switched off.
- Alarm weekdays only – The Alarm will only fire Monday through Friday. It is not reset.

Now move to "Set alarm time:" using the channel UP control. The current alarm time will be displayed on line 2 of the display. Using the volume UP and DOWN control adjust the alarm time to the required setting. If you do not wish to put the radio into sleep mode at this stage then use the channel UP/DOWN control to move away from the "Set alarm time:" option and press the Menu button. If you press the Menu button whilst in the "Set alarm time:" option and the Alarm is set to anything except off then the radio will enter Sleep mode and display the alarm on line 2 for a two-line LCD or on line 4 for a four-line LCD.

Note: On the rotary encoder version of the radio turning the volume control faster up or down will increment the timer hours.

Note: Sleep mode can only be exited by pressing the Menu button.

The alarm function uses the **/var/lib/radiod/alarm** file which will contain the current alarm type and time. The format is **t:hh:mm** where t is type (0=off, 1=on, 2=repeat, 3=weekdays only) and hh:mm is hours and minutes (24 hour clock). You do not need to change the contents of this file.

PLEASE NOTE THAT THE ALARM RELIES UPON THE SELECTED RADIO STREAM TO BE AVAILABLE WHEN THE ALARM WAKES UP. THIS CANNOT BE GUARANTEED AS THE STATION FEED MAY BE OFF AIR OR THERE IS A PROBLEM WITH THE INTERNET CONNECTION. YOU SHOULD NOT THERFORE RELY SOLEY ON THIS ALARM FUNCTION IF YOU HAVE AN IMPORTANT APPOINTMENT OR A PLANE OR TRAIN TO CATCH FOR EXAMPLE. ALSO SEE DISCLAIMER ON PAGE 80.

Using the Alarm and Timer functions together

The Alarm and Timer functions can be used together. For example you want to set your radio to a 30 minute snooze time before going to sleep and to sound the alarm in the morning. Simple set the Timer to the required elapse time and then set the alarm as described in the previous section. Press the Menu button and the timer will be seen counting down followed by the alarm time on line 4 or line 1 for the four-line and two-line LCD respectively.

Music Player Clients

MPD is designed around a client/server architecture, where the clients and server (MPD is the server) interact over a network. A large number of graphical and web based clients are available for MPD and are to numerous to mention here. Please see the following link for further information on MPD clients: http://mpd.wikia.com/wiki/Clients. The main client used in this project is MPC.

Using the MPC client

Everything you should normally wish to do can be done using the radio. However there may be occasions that you wish to test or control music selection, volume etc. using MPC. It is also useful for diagnosing Music Player Daemon problems.

Log into the Raspberry PI using the console or SSH login. To start playing music run:

```
$ mpc play
```

To see a list of all available commands, run:

```
$ mpc help
```

Here are some frequently used **mpc** commands:

MPC command	Description
mpc	Displays status (**mpc status** also does the same)
mpc current	Displays currently playing station or track
mpc next	Play next song
mpc prev	Play previous song
mpc play n	Play station or track where n is the track or station number
mpc volume 75	Set volume to 75%
mpc stop	Stop playing
mpc random <on|off>	Toggle shuffling of songs on or off
mpc repeat <on|off>	Toggle repeating of the playlist
mpc clear	Clear the playlist
mpc consume <on|off>	When playing tracks remove from the playlist once played
mpc playlist	List loaded radio stations or streams
mpc listall	List all songs in the music directory

Shutting down the radio

You can simply switch the power off. This doesn't appear to harm the PI at all. However if you want a more orderly shutdown then press the menu button for at least three seconds. This will stop the MPD daemon and issue a shutdown request to the Raspberry PI. Wait <u>at least</u> another ten seconds and then power off the Radio.

Note: This function is only available in the Rotary encoder version of the radio from version 3.2 onwards.

Creating and Maintaining Playlist files

Overview media stream URLs

A deep understanding of this section is not necessary but can help in creating playlists. At first the whole business of how music streams are provided can be quite confusing. The URLs on a radio station web page can be of different types for example:

1. A URL pointing to a PLS playlist file(Shoutcast Play List)
2. A URL pointing to a M3U playlist file (MPEG3 URL)
3. A URL pointing to an ASX playlist file (Advanced Stream Redirector)
4. A URL which is an actual stream such as MP3 (MPEG 3) or AAC (Advanced Audio Coding)

1, 2 and 3 are so called redirector URLs and point to a playlist file containing one or more URLs to the radio stream(s) itself. The *create_playlists.py* program tries to figure out what type of URL that it is and create a playlist from it. This is the facility you should use rather than trying to create your own playlists which can be quite time consuming.

PLS file format

A good place to start is the following Wikipedia article:

http://en.wikipedia.org/wiki/PLS_(file_format)

A PLS playlist file does not contain any music files itself, but rather points to music files stored elsewhere. The PLS file format is often used to play Internet radio streams, for example, if you want to play a radio stream from Shoutcast, you can copy the PLS file URL of the station from the site and play it in a desktop media player like Winamp. A PLS file will be similar to below

```
[playlist]
NumberOfEntries=2
Version=2
File1=http://206.217.213.16:8430
Title1=Blues Radio UK
Length1=-1
File2=http://205.164.62.13:8030
Title2=Absolute Blues Hits
Length2=-1
```

The PLS file must always start with the **[playlist]** statement. The **NumberOfEntries** statement must match the number of streams defined in the PLS file (Two in the above example). Set the **Version** number always to 2.

There must be a File*n*, Title*n* and Length*n* where *n* is the entry number.

M3U Files

M3U stands for MPEG3 URL. The following Wikipedia article explains the M3U file format:

http://en.wikipedia.org/wiki/M3U

An example M3U file is shown below:

radio10.m3u playlist file

```
#EXTM3U
#EXTINF:-1, Radio 10 Gold NL
http://icecast.streaming.castor.nl:80/radio10
```

These playlist files must have the m3u file extension. i.e. **<filename>.m3u**

The first line is the header and must be #EXTM3U. The second line is #EXTINF: and is information about the radio stream. The -1 means unlimited play length. This is followed by a comma and then the name of the radio station (*Radio 10 Gold* in this case). The third line is the URL (icecast in this case) for the radio stream. More than one radio stream may be defined in the m3u file. Simple add extra #EXTINF and URL lines for each radio stream.

ASX file

The Advanced Stream Redirector (ASX) format is a type of XML metafile designed to store a playlist of Windows Media files for a multimedia presentation. An example ASX file is shown below.

```
<ASX version="3.0">

<ABSTRACT>http://www.bbc.co.uk/iplayer/radio/bbc_radio_bristol/</ABSTRACT>
        <TITLE>BBC Bristol</TITLE>
        <AUTHOR>BBC</AUTHOR>
        <COPYRIGHT>(c) British Broadcasting Corporation</COPYRIGHT>
        <MOREINFO
HREF="http://www.bbc.co.uk/iplayer/radio/bbc_radio_bristol/" />
        <PARAM NAME="HTMLView"
VALUE="http://www.bbc.co.uk/iplayer/radio/bbc_radio_bristol/" />
        <Entry>
                <ref href="mms://wmlive-
nonacl.bbc.net.uk/wms/england/lrbristol?BBC-
UID=1523a2f20f858eb0ba0a4385918e6433df886bb650e021b4446ff486f8204e3a&amp;SSO
2-UID=" />
        </Entry>
</ASX>
```

Direct stream URLs

These URLs tend to end with .mp3 or _SC or AAC etc. However there are others. For example:

http://mp3.streampower.be/radio1-high.mp3
http://7639.live.streamtheworld.com:80/977_MIX_SC

You can determine if a URL is a direct radio stream by using the **wget** program.

```
# cd /tmp
# wget http://mp3.streampower.be/radio1-high.mp3
--2014-03-14 13:08:10--  http://mp3.streampower.be/radio1-high.mp3
Resolving mp3.streampower.be (mp3.streampower.be)... 80.200.255.61
Connecting to mp3.streampower.be (mp3.streampower.be)|80.200.255.61|:80...
connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [audio/mpeg]
Saving to: `radio1-high.mp3'
```

```
     [                                                            <=>              ]
 365,281      15.8K/s
```

If **wget** doesn't exit and you see the <=> characters moving backwards and forwards then it is a URL to the radio stream itself. You will also see *Length: unspecified* in the output. Press control -C to exit **wget**. Remove the file that **wget** created (/tmp/radio-high.mp3 in this case).

Creating playlists

Since version 2.3 of the radio daemon the *create_playlists.py* program is used to create playlists in the **/var/lib/mpd/playlists** file.

If you are an existing user of the Radio program please read the section on the playlists directory on page 51.

You should read and understand the previous section before using this program. The directories and files used by the *create_playlists.py* program are shown in the following table:

Table 6 Playlist files and directories

Name	Type	Description
/home/pi/radio/station.urls	File	Initial distribution file containing sample stations
/var/lib/radiod/stationlist	File	The file containing the users list of radio stations
/home/pi/radio/playlists	Directory	The directory containing any user playlist files
/tmp/radio_stream_files	Directory	Temporary work directory used during processing
/var/lib/mpd/playlists	Directory	The Music Player Daemon Playlist directory

From a user point of view only the **/var/lib/radiod/stationlist** file and the **/home/pi/radio/playlists** directory are important**.**

During installation you should have run the *create_playlists.py* at least once. This powerful little program does the following:

1. Copies the /**home/pi/radio/station.urls** file to the **/var/lib/radiod/stationlist** file. It does this only the first time that it runs. The **station.urls** file will not normally be used again.
2. Copies all playlists it finds in the **/home/pi/radio/playlists** directory to the **/var/lib/mpd/playlists** directory.
3. It reads each entry in the **/var/lib/radiod/stationlist** file.
4. If it comes across a (<playlist name>) definition it creates a new playlist.
5. If the line read contains a station definition then it determines if the URL in the station entry is a direct radio stream (.mp3, AAC etc) or if it is a redirect URL (.pls, .asx or .m3u).
6. If it is a direct radio stream URL it creates a playlist in PLS format and in the **/tmp/radio_stream_files** directory.
7. If it is an ASX URL (.asx) it reads the contents of the ASX file and uses them to create a playlist in PLS format in the **/tmp/radio_stream_files** directory.
8. If it is a redirect URL (.pls or .m3u) gets the file pointed to in the URL and from the contents of this file it creates an entry in PLS format in the **/tmp/radio_stream_files** directory.
9. It prompts you if you wish to remove all removes all the old playlists from the **/var/lib/mpd/playlists** directory.

10. It finally copies all files in the **/tmp/radio_stream_files** directory to the **/var/lib/mpd/playlists** directory.

The program itself is very easy to use. Just run it as sudo or root user in the /home/pi/radio directory:

```
$ sudo ./create_playlists.py
```

This will create a set of playlist files in the **/var/lib/mpd/playlists** directory. The **/var/lib/radiod/stationlist** file contains a list of entries describing the radio station to be loaded.

To create a log file of the program run the following:

```
$ sudo ./create_playlists.py | tee playlist.log
```

You can examine the playlist.log file to see what actions the create_playlist.py program carried out and if there were any errors.

If you are running the program for a second time it will ask if you wish to delete any old

```
There are 8 old playlist files in the /var/lib/mpd/playlists/ directory.
Do you wish to remove the old files y/n: y
```

Normally answer 'y' unless you don't wish to remove the old files. Note that old playlists files with the same name as the new ones will always be overwritten. Any custom play lists should be copied to the **/home/pi/radio/playlists** directory anyway. They will then always be copied back to the **/var/lib/mpd/playlists directory** by the *create_playlists.py* program.

If you want to avoid the above prompt then there are two other parameters that you may use.

--delete_old	Delete old playlist files in the MPD playlist directory
--no_delete	Don't delete old playlist files in the MPD playlist directory

Example:

```
$ sudo ./create_playlists.py --no_delete
```

Finally there is a help parameter:

```
$ sudo ./create_playlists.py --help
```

The stationlist file

The **/var/lib/radiod/stationlist** file is the file that should be maintained by you to create playlists. When this *create_playlists.py* program is first run it copies the distribution file **station.urls** to the **/var/lib/radiod/stationlist** file. You may then modify the **/var/lib/radiod/stationlist** file.

The format is:	**(<playlist name>)**
Example:	**(BBC Radio Stations)**

The above will create a playlist called **BBC_Radio_Stations.pl** and will contain

Now add or remove radio station definitions in the **stationlist** file. The first statement in the station definition is the name of the playlist in brackets:

The format is: **[<title>] http://<url>**
Example: **[BBC Radio 4 extra] http://www.bbc.co.uk/radio/listen/live/r4x.asx**

After modifying the stationlist file run the *create_playlists.py* program to create the Music Player Daemon playlists.

The *create_playlists.py* program creates file names using the title so in the above example the file will be called *BBC_Radio_4_extra.pls* (The AS format will be converted to PLS format). If this file already exists the new file will be renamed as *BBC_Radio_4_extra[1].pls.* If that file also exists the program will keep incrementing the number in brackets until the name is unique. This is one of the reasons the files are created in a temporary directory first. Try to create unique titles in the **stationlist** file to avoid this.
If no title is found then the site name will be used with dots turned into underscores. For example if the following is defined without a title and no title is found:

[]http://bbc.co.uk/radio/listen/live/r1.asx

This will create a playlist file called *bbc_co_uk.pls*. If this already exists it will be renamed to *bbc_co_uk[1].pls.* If that file also exists the program will keep incrementing the number in brackets until the name is unique. It is always better to define a title.

The playlists directory

There is a playlists directory in **/home/pi/radio** directory. This is meant for existing users to store their existing playlist files or to store playlist files (.pls or m3u) or playlist files which have been manually created. When the *create_playlists.py* program is run, the first thing it does is to copy the playlist files in the **/home/pi/radio/playlists** directory to the **/var/lib/mpd/playlists**.

Existing users should backup all of their playlist files and move all existing playlist files to the **/home/pi/radio/playlists** directory.

```
# cd /tmp
# tar -cvf playlists.tar /var/lib/mpd/playlists/*
# mv /var/lib/mpd/playlists/* /home/pi/radio/playlists/.
```

Run the *create_playlists.py* program to create the new playlists in the MPD playlist directory.

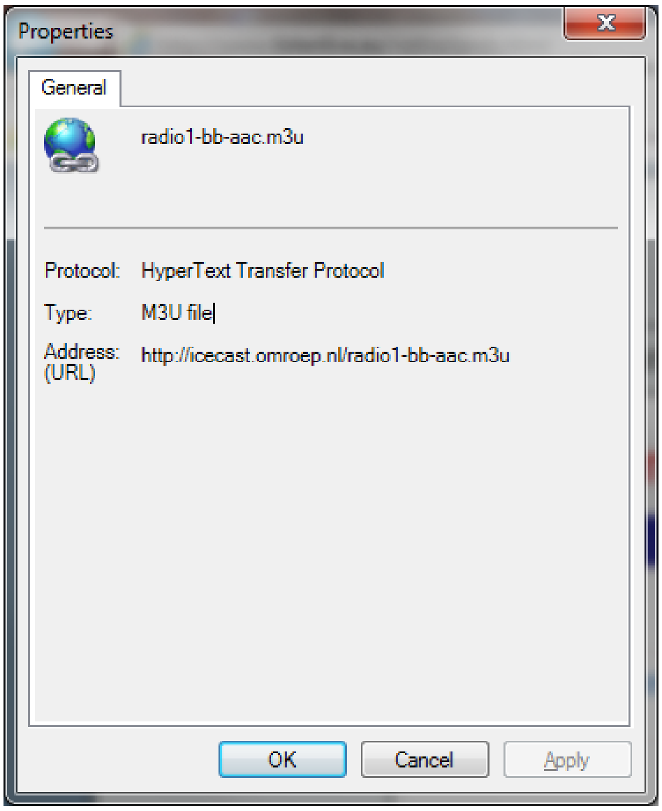
Note: Existing users should try to migrate their existing URLs to the /var/lib/radiod/stationlist format. This will be much quicker and easier than trying to maintain the playlist files directly.

Radio stream resources on the Internet

There are a lot of resources on the Internet how to find PLS and M3U files so simply search for "PLS or M3U files" through the search machine of your choice. Below are some good sources of radio streams around the world.

http://www.listenlive.eu
http://www.internet-radio.com
http://www.radio-locator.com
http://bbcstreams.com/

To copy a URL open the web page in any browser on a PC and right click on the URL. Select properties from the drop down list. For internet explorer will show a window similar to the following will be displayed:



radio1-bb-aac.m3u

Protocol: HyperText Transfer Protocol

Type: M3U file

Address: (URL) http://icecast.omroep.nl/radio1-bb-aac.m3u

Copy and paste the URL into the **/var/lib/radiod/stationlist** file. Add the title in square brackets as shown in the previous section. Other browsers may provide options such as 'copy link'or 'save link as'. This is browser dependant.

Installing the Web interface

MPD has several web clients. See the following link: http://mpd.wikia.com/wiki/Clients. The one used in this example is called snoopy is released with version 1.16 onwards of the radio software.

Install Apache

Install Apache the web server. Make sure that the system is up to date with the following command.

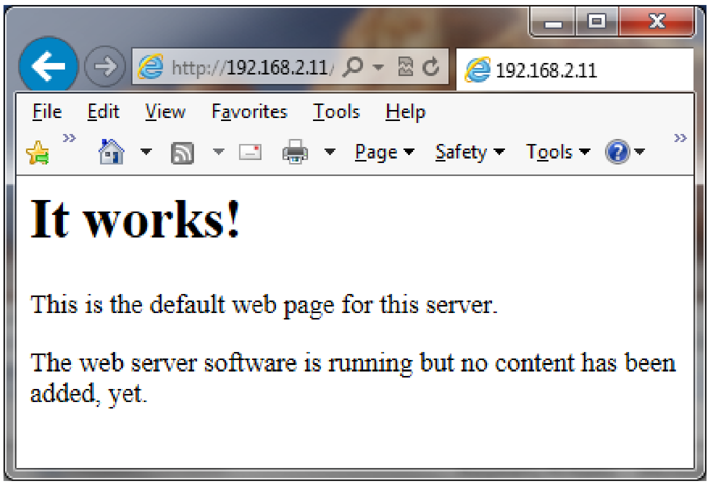
```
# apt-get update
```

Now install Apache and the PHP libraries for Apache.

```
# apt-get install apache2 php5 libapache2-mod-php5
```

Test the Apache web browser

Point your web browser at the IP address of the Raspberry PI. For example: http://192.168.2.11 . You should see the following display.



It works!

This is the default web page for this server.

The web server software is running but no content has been added, yet.

Install the Web Browser server pages

It is now necessary to install the web pages for the Radio. Download the radio web pages from http://www.bobrathbone.com/raspberrypi/source/pi_radio_web.tar.gz

```
# wget http://www.bobrathbone.com/raspberrypi/source/pi_radio_web.tar.gz
```

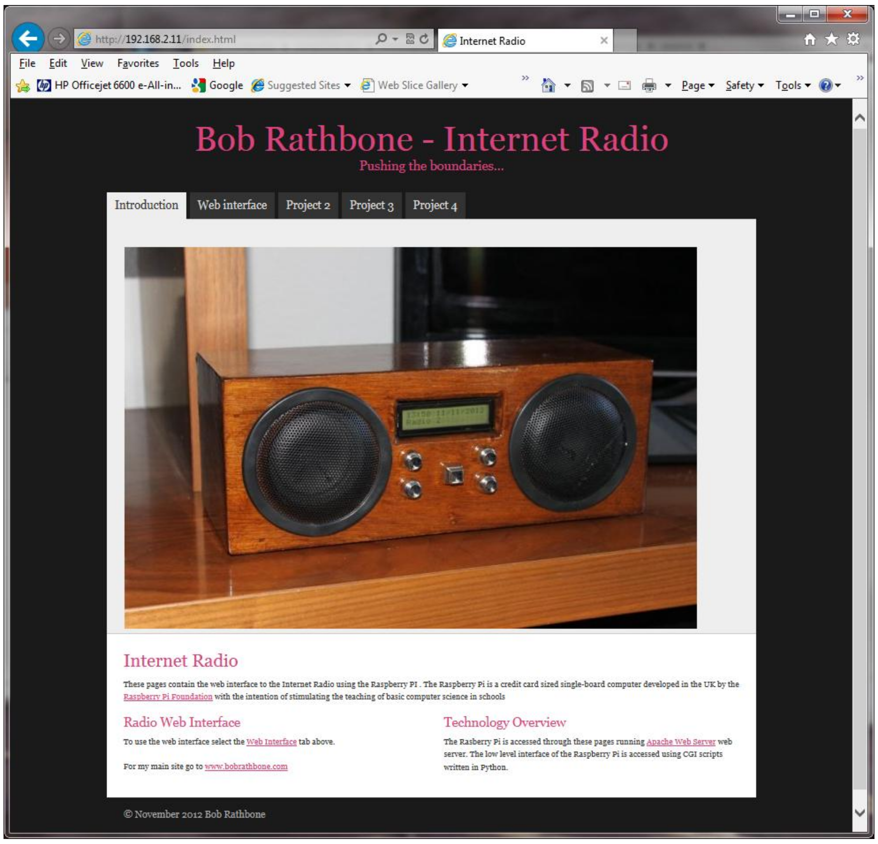
Copy this to the **/home/pi/radio** directory. Now install the pages with the following commands:

```
# cd /
# tar -xvf /home/pi/radio/pi_radio_web.tar.gz
```

This will install the radio pages in the **/var/www** directory and the CGI scripts in **/usr/lib/cgi-bin** directory.

Start the radio web interface

Point your web browser at the IP address of the Raspberry PI. For example: http://192.168.2.11 . You should see the following display:


http://192.168.2.11/index.html
Internet Radio
File Edit View Favorites Tools Help
HP Officejet 6600 e-All-in... Google Suggested Sites Web Slice Gallery
Page Safety Tools
Introduction
Web interface
Project 2
Project 3
Project 4

Bob Rathbone - Internet Radio

Pushing the boundaries...

Internet Radio

These pages contain the web interface to the Internet Radio using the Raspberry PI . The Raspberry Pi is a credit card sized single-board computer developed in the UK by the Raspberry Pi Foundation with the intention of stimulating the teaching of basic computer science in schools

Radio Web Interface

To use the web interface select the Web Interface tab above.

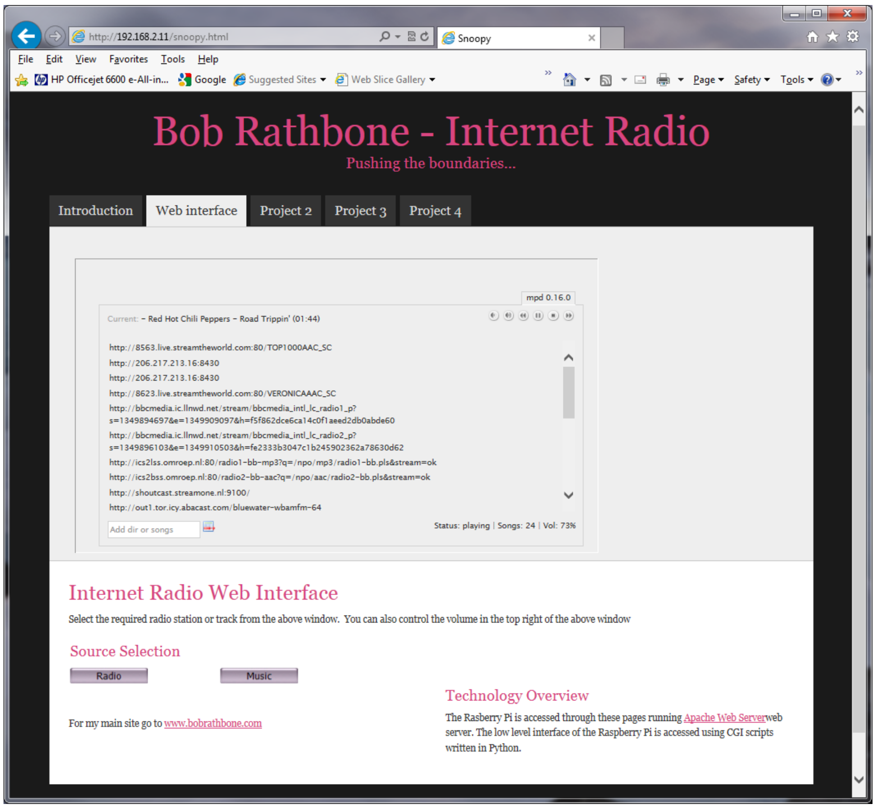
For my main site go to www.bobrathbone.com

Technology Overview

The Rasberry Pi is accessed through these pages running Apache Web Server web server. The low level interface of the Raspberry Pi is accessed using CGI scripts written in Python.

© November 2012 Bob Rathbone

Now click on the 'Web interface tab'. If the radio software is running you will see the following:


http://192.168.2.11/snoopy.html
Snoopy
File Edit View Favorites Tools Help
HP Officejet 6600 e-All-in... Google Suggested Sites Web Slice Gallery
Page Safety Tools
Introduction
Web interface
Project 2
Project 3
Project 4
mpd 0.16.0
Add dir or songs
Radio
Music

Bob Rathbone - Internet Radio

Pushing the boundaries...

```
Current: - Red Hot Chili Peppers - Road Trippin' (01:44)

http://8563.live.streamtheworld.com:80/TOP1000AAC_SC
http://206.217.213.16:8430
http://206.217.213.16:8430
http://8623.live.streamtheworld.com:80/VERONICAAAC_SC
http://bbcmedia.ic.llnwd.net/stream/bbcmedia_intl_lc_radio1_p?
s=1349894697&e=1349909097&h=f5f862dce6ca14c0f1aeed2db0abde60
http://bbcmedia.ic.llnwd.net/stream/bbcmedia_intl_lc_radio2_p?
s=1349896103&e=1349910503&h=fe2333b3047c1b245902362a78630d62
http://ics2lss.omroep.nl:80/radio1-bb-mp3?q=/npo/mp3/radio1-bb.pls&stream=ok
http://ics2bss.omroep.nl:80/radio2-bb-aac?q=/npo/aac/radio2-bb.pls&stream=ok
http://shoutcast.streamone.nl:9100/
http://out1.tor.icy.abacast.com/bluewater-wbamfm-64

Status: playing | Songs: 24 | Vol: 73%
```

Internet Radio Web Interface

Select the required radio station or track from the above window. You can also control the volume in the top right of the above window

Source Selection

For my main site go to www.bobrathbone.com

Technology Overview

The Rasberry Pi is accessed through these pages running Apache Web Serverweb server. The low level interface of the Raspberry Pi is accessed using CGI scripts written in Python.

Click on any station on the list to select a station. The Radio and Music buttons select the source.

Mounting a network drive

It is very likely that you may have your music on a shared network drive and want to play the music through the radio. This is possible from version 1.6 onwards of the radio software. There are two main types of network drive protocols used by Raspian Wheezy on the Raspberry Pi namely:

- CIFS – Common Internet File System
- NFS – Network File System

There is a third type of network file protocol called SMB (Server Message Block – Microsoft) but is replaced by CIFS in the Raspberry PI. Your PC will be using SMB most probably but CIFs also supports this interface. The steps to mount the network drive are as follows:

1. Find out the IP address of your network drive.
2. Create and test the mount command using either NFS or CIFS.
3. Copy the mount command to **/var/lib/radiod/share** file.
4. In the Radio menu select "Music Library" as the source and press "Menu" again to load
5. Update the playlists to include the files on the new share (Network drive).

This procedure assumes that you already have your Network Drive configured and working with your PC and can play music via the PC. In the examples below a Synology Network Drive was used with a volume called Volume1 with a directory called "music". The IP address for the Synology Network drive used was 192.168.2.6.

First stop the Radio software when creating and testing the mount command.

Don't configure **/etc/fstab** to do the mount of the network drive. Although this is the usual way of mounting shares however the radio program needs total control of the mount and un-mount process.

The general syntax for the mount command is as follows:

```
mount -t <type> -o option1,option2,… <remote IP address and directory>
<mount point>
```

Where: <type> is either **nfs** or **cifs**.
-o option1,option2 are the mount options.
<remote IP address and directory> Is the IP address and music directory path
<mount point> This will always be /share for this program

Finding the IP address of the network drive

Only general guidance can be given here. Nearly all network drives have a web interface. The IP address was almost certainly provided from DHCP in your home router. The IP address will be the IP address of the Web Interface. Look at your network drive documentation for further information.

The CIFS mount command

The following example mount command assumes that you have a guest user configured with password 'guest'. Adapt the command as required.

```
mount -t cifs -o username=guest,password=guest  //192.168.2.6/music /share
```

The share directory is created when you first run the Radio program (v1.6 onwards) so there is no need to create it. If the command was successful you should be able to display the music from the network drive. Go to section called *Display the share directory* on page 56.

The NFS mount command

The following NFS mount example assumes the NFS protocol has been configured for the music directory.

```
mount -t nfs -o ro,nolock 192.168.2.6:/volume1/music /share
```

A few things to note here; the NFS mount command uses the volume name (Volume1), The CIFS mount command doesn't. The second thing is that the IP address and remote directory are separated by a colon (:). If the command was successful you should be able to display the music from the network drive.

Display the share directory

If the mount was successful using either CIFS or NFS you should be able to display the **/share** directory with **ls**.

```
# ls -la /share
total 576
drwxrwxrwx 144 1024 users 4096 Feb  1 12:07 .
drwxr-xr-x  23 root root  4096 May 15 10:48 ..
drwxrwxrwx   3 1024 users 4096 Feb  1 12:05 Adriano Celentano
drwxrwxrwx   3 1024 users 4096 Feb  1 12:07 Afric Simone
drwxrwxrwx   3 1024 users 4096 Feb  1 12:07 Al Martino
drwxrwxrwx   4 1024 users 4096 Feb  1 12:05 America
drwxrwxrwx   3 1024 users 4096 Feb  1 12:06 Aphrodite's Child
```

Un-mounting the /share directory

To un-mount the share directory use the **umount** command (not unmount).

```
# umount /share
```

Copy the mount command to the configuration

Once the mount command is working copy it to the **/var/lib/radiod/share** file.
For example for the CIFS mount command.

```
# echo "mount -t cifs -o username=guest,password=guest  //192.168.2.6/music
/share" > /var/lib/radiod/share
```

Load the music library

Now run the radio program. The radio stations will be loaded. Cycle through the menu until **Input Source**: is displayed. Press the channel up or down buttons to select **Music Library**. Now press the **Menu** button. The program loads whatever playlists it has in its database, and will most likely be only those from the USB stick if installed. However the *playlist* for the new share files are not yet in the MPD database. The playlist needs to be updated in the following section.

Update the playlists for the new share

Select Music Library Now cycle through the menu until **Menu Selection:** is displayed. Press the channel up or down buttons until the **Update list:No** is displayed. Use the Volume buttons to toggle the display to **Update list:Yes**. Now press the Menu button. This will cause the MPD database to be cleared and updated from all the files loaded in the **/var/lib/mpd/music** directory including the new share. This can take some time (Several minutes) if the Network Drive contains a large amount of music files. During this process the Radio program will ignore any button depressions and you will see the first **Initialising** (Library) and then **Updating** (Library).

Disabling the share

To disable the share simply put a hash character (#) at the beginning of the line in the **/var/lib/mpd/share** file as shown in the example below. Alternatively remove the share file altogether.

```
#mount -t cifs -o username=guest,password=guest  //192.168.2.6/music /share
```

Further information

For your information if you display the **/var/lib/mpd/music** directory you will see two soft links to the **/share** and **/media** directories for the network drive and USB stick respectively.

```
# ls -la /var/lib/mpd/music/
total 8
drwxr-xr-x 2 root root  4096 May 19 11:17 .
drwxr-xr-x 4 mpd  audio 4096 May 16 19:02 ..
lrwxrwxrwx 1 root root     6 May 19 11:17 media -> /media
lrwxrwxrwx 1 root root     6 May 19 11:17 share -> /share
```

These links are created automatically by the Radio program. If these are missing they can be re-created with the ln –s command.

```
# cd /var/lib/mpd/music
# ln -s /media
# ln -s /share
```

This shouldn't normally be necessary as the links are created by the program when it creates the media and share mount points.

Miscellaneous

Configuring the Adafruit LCD backlight colours

Some Adafruit displays such as the **rgb-negative Adafruit LCD** allow changing the colour of the backlight. This isn't configurable at the moment and can only be achieved by modifying the code in the **ada_radio.py** program file.

You can change the backlight colour with the **lcd.backlight(lcd.color)** command where color is RED, YELLOW, BLUE etc. The colours are defined in the **ada_lcd_class.py** file.
Modify the **test_ada_lcd.py** program first as shown below to test the colours you want to use.

```
while True:
        if lcd.buttonPressed(lcd.MENU):
                print("Menu button")
                lcd.backlight(lcd.RED)
                lcd.line2("Menu button")
        elif lcd.buttonPressed(lcd.LEFT):
                print("Left button")
                lcd.backlight(lcd.BLUE)
                lcd.line2("Left button")
        elif lcd.buttonPressed(lcd.RIGHT):
                print("Right button")
                lcd.backlight(lcd.GREEN)
                lcd.line2("Right button")
        elif lcd.buttonPressed(lcd.UP):
                print("Up button")
                lcd.line2("Up button")
        elif lcd.buttonPressed(lcd.DOWN):
                print("Down button")
                lcd.backlight(lcd.YELLOW)
                lcd.line2("Down button")
```

Now modify the main loop in ada_radio.py" (line 99 onwards) to use the colours. For example if you want to change the background colour to RED during shutdown then modify the code as shown below.

```
                        # Shutdown command issued
                        if display_mode == radio.MODE_SHUTDOWN:
                                lcd.backlight(lcd.RED)
```

Note: Always use the American spelling 'color' in all commands and not the British spelling 'colour'.

Using the Adafruit backlit RGB LCD display

The Adafruit backlit RGB LCD has three LED backlights (Red, Blue and Green) which can either be switched on individually or in various combinations together as shown in the table below:

Table 7 Adafruit backlit RGB display wiring

Red (Pin 16)	Green (Pin 17)	Blue (Pin 18)	Colour	Diodes required
0	0	0	Off	0
0	0	1	Blue	0
0	1	0	Green	0
0	1	1	Light Blue	2
1	0	0	Red	0
1	0	1	Purple	2
1	1	0	Yellow	2
1	1	1	White	3

So to use all of the above combinations would require a single pole 8 way rotary switch or logic. The first switch position is off.

- Wire pin 16 (Red) of the LCD to switch position 2.
- Wire pin 17 (Blue) of the LCD to switch position 3
- Wire pin 18 (Green) of the LCD to switch position 4
- Wire pin 17 and 18 via two diodes to pin 5 to give the colour light blue
- Do the same for the other two colour combinations
- Wire pin 16, 17 and 18 to pin 8 via three diodes to give the colour white

Troubleshooting

LCD screen not working

Check that the wiring conforms to the *wiring list* on page 12. Make sure that pin 3 is grounded (0V) to give maximum contrast. If you are using the Adafruit LCD plate the make sure that you are running the **ada_radio.py** program and not one of the other programs (See Table 1 on page 12).

Run the *lcd_test.py* program to see if the LCD displays anything. This runs independently of any other software and can be used stand alone.

The LCD displays hieroglyphics

If you have a revision 1 Raspberry PI board you need to amend the *lcd_class.py* code as shown in the section called *The LCD Class* on page 29. If you are using a revision 2 board do not amend the *lcd_class.py* code as it is designed to work with this board. This problem has also been experienced with faulty LCD hardware particularly when re-booting the Raspberry PI.

Check the wiring conforms to the *wiring list* on page 12. In particular check the data lines to pins 11, 12, 13 and 14 (See *LCD* Module Wiring on page 15). Retest the LCD using the *lcd_test.py* program.

LCD backlight not working

Check that pins 15 and 16 of the LCD display have +5V and 0V(GND) respectively. See *LCD* Module Wiring on page 15.

LCD only displays dark blocks on the first line

This is normal when the raspberry PI starts up. The display should work with the *lcd_test.py* program. If the *lcd_test.py* program still doesn't display anything then check that the wiring conforms to the *wiring list* on page 13. If you are using the Adafruit LCD plate the make sure that you are running the **ada_radio.py** program and not one of the other programs (See Table 1 on page 12).

MPD fails to install

During installation of MPD some files return a 404 error (Not found) the following message is seen.

```
Unable to fetch some archives, maybe run apt-get update or try with -fix-
missing?
```

This is due to that an update was not previously carried out as shown in the section called *SD card creation* on page 22. Perform the update and upgrade as shown and re-install MPD and MPC.

Music Player Daemon won't start

The MPD daemon logs to the **/var/log/mpd/mpd.log** file. Examine this file for errors. The MPD daemon is dependant on good PLS files so check that these are correct as described in the section called *Creating and Maintaining* Playlist files on page 47.

The MPD may display a socket error

When starting the MPD daemon the following message is seen:

```
Starting Music Player Daemon: mpdlisten: bind to '[::1]:6600' failed: Failed
to create socket: Address family not supported by protocol (continuing
anyway, because binding to '127.0.0.1:6600' succeeded)
```

If this message is seen in the MPD log file this is simply because IP version 6 (IPv6) isn't installed so the message doesn't affect operation of the MPD.

To prevent it from happening configure the *bind_to_address* parameter in the **/etc/mpd.conf** file to "any" as explained in the section called *Configure the MPD daemon* on page 32.

ImportError: No module named mpd

The following message is seen when attempting to start the radio:

```
Traceback (most recent call last):
  File "/home/pi/radio/ada_radio.py", line 33, in <module>
    from radio_class import Radio
  File "/home/pi/develop/pi/radio/radio_class.py", line 26, in <module>
    from mpd import MPDClient
ImportError: No module named mpd
```

This only happens from version 3.3 onwards which now uses the python-mpd library.
Run the following command:

```
apt-get python-mpd
```

Restart the radio program.

PLS files won't load using MPC

When attempting to load a PLS file (for example ukblues.pls) using the mpc client you see the following message:

```
# mpc load ukblues.pls
Loading: ukblues.pls
Error: no such playlist
```

This is due to permissions on the files by not copying them to **/var/lib/mpd/playlists** directory using sudo or as root user. This can be seen by using the **'ls –la'** command to display the permiisions.

```
pi@raspberrypi:~$ ls -la /var/lib/mpd/playlists/ukblues.pls
-rw-r----- 1 root root 105 Oct  5 10:44 /var/lib/mpd/playlists/ukblues.pls
```

The **-rw-r-----** string means the other users (including pi) cannot read this file. You may need to learn about file permissions if you don't already know. The problem can be solved by setting the permissions for "other" to read:

```
pi@p raspberrypi:~$ sudo chmod o+r /var/lib/mpd/playlists/*.pls
pi@ raspberrypi:~$ ls -la /var/lib/mpd/playlists/ukblues.pls
-rw-r--r-- 1 root root 105 Oct  5 10:44 /var/lib/mpd/playlists/ukblues.pls
```

This should solve the problem as the **rw-r--r--** string means the other users (including pi) can read the file.

Cannot start the radio daemon with sudo

If the radio daemon only starts as user root; this is due to file permission problems with the files in the **/var/lib/radiod** directory. To correct it carry out the following instruction:

```
sudo chmod -R 766 /var/lib/radiod
```

The radio daemon should now start as sudo without further problem.

The MPD daemon complains about the avahi daemon

The following message is seen in the **/var/log/mpd/mpd.log** file

```
Apr 10 15:37 : avahi: Failed to create client: Daemon not running.
```

Change the *zeroconf_enabled* parameter in the **/etc/mpd.conf** file to “no” as explained in the section called *Configure the MPD daemon* on page 32. The *avahi* daemon is used to configure systems without a network connection but is not enabled by default. It is not required for this design.

Buttons seem to be pressing themselves

The symptoms are that it looks like buttons are generating their own signals i.e. they appear to being continually pressed although they are not being operated. In particular the MENU button displays this problem. This is because the inputs are “floating”. All inputs for the button operated radios (Not Adafruit plate) need to be pulled down to ground using a 10K resistor for version 1 boards. Newer version 2.0 boards have inbuilt pull-up/pull-down resistors. Version 2.0 onwards of the radio software enables the internal pull-down resistors so doesn’t require external resistors. Use the very latest version of the software to eliminate this problem. Use the **test_switches.py** software to test the buttons.

Radio daemon doesn’t start or hangs

This is almost certainly a problem with either the MPD daemon or failed internet connection.”Check the network connection and run installation tests on the MPD daemon. Occasionally a bad PLS file can cause this problem. You can check that your Raspberry PI has an internet connection with the *ip addr* command. The example below shows interface *eth0* connected as IP 192.168.2.22.

```
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP qlen 1000
    link/ether b8:27:eb:fc:46:15 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.22/24 brd 192.168.2.255 scope global eth0
```

Stream decode problems

The radio may display a message similar to the following:

```
ERROR: problems decoding http://173.244.194.212:8078
```

This is due to an invalid URL (In the above example this is http://173.244.194.212:8078) in one of the PLS files.

Locate the offending URL in the play list file in the **/var/lib/mpd/playlists** directory. Either correct the radio stream URL or remove it all together.

Also check that the file URL is not the pointer to the PLS file (See section Creating and Maintaining Playlist files on page 47.

Cannot mount remote network drive

There are just too many possibilities to cover all of these here. However a few common problems are covered here:

Error: mount error(115): Operation now in progress
Cause: Most likely an incorrect IP address

Error: NFS mount hangs
Cause: Most likely an incorrect IP address

Error: mount.nfs: access denied by server while mounting <ip address>:/music
Cause: The volume name is missing – for example /volume1/music

Error: mount error(16): Device or resource busy
Cause: The share mount directory is in use because a mount has already been done. Run the umount command.

Error: mount error(2): No such file or directory
Cause: The path specified in the mount doesn't exist

Error:
mount.nfs: rpc.statd is not running but is required for remote locking.
mount.nfs: Either use '-o nolock' to keep locks local, or start statd.
mount.nfs: an incorrect mount option was specified
Cause:
You need to include the "–o noclock" option.

If the error isn't in the above list then search the web for suggestions.

Button or Rotary encoder problems

Use the *test_switches.py* or test*_rotary_class.py* to test the push buttons or rotary encoders respectively.

Rotary switches not working

Check wiring in particular the common pin must be connected to ground (and not 3.3 volts). Run the test_rotary_class.py to test the switches.

If you see a message similar to the following then you are using the wrong GPIO libraries most likely because you haven't installed the correct version of Raspian Wheeze on the SD card as shown on page 22:

```
Traceback (most recent call last):
  File "test_rotary_class.py", line 38, in <module>
    volumeknob =
RotaryEncoder(LEFT_SWITCH,RIGHT_SWITCH,MUTE_SWITCH,callback)
  File "/home/pi/Projects/radio/rotary_class.py", line 39, in __init__
    GPIO.add_event_detect(self.pinA, GPIO.FALLING,
callback=self.switch_event)
AttributeError: 'module' object has no attribute 'add_event_detect'
```

Install the latest version of Raspian Wheeze as shown on page 22.

If you are running an older version Raspian Wheeze you can connect to the Internet and run the two following commands as user root.

```
# apt-get update
# apt-get upgrade
```

This can take some time. Reboot once the upgrade is finished.

Volume control not working with USB speakers

This is hardware dependant. Not all USB hardware and drivers work with mixer type "hardware". If this problem is being experienced try setting the **mixer_type** parameter to "software". Edit the /etc/mpd.conf file and change the mixer type to software.

```
mixer_type                      "software"
```

Remove the # at the beginning of the line to enable software mixing and save the file. Restart the radio software.

Note: This solution was provided by one of the constructors and is untested by the author.

The message "Check playlists" is displayed

The message "Check playlists" is displayed on the LCD screen for certain radio stations. This will only be seen in versions 3.3 and 3.4. The reason is that certain radio stations are not including the station name in their stream. This is also why the web interface doesn't display a station name with these radio streams.
Upgrade to version 3.5 and above to cure this problem or complain to the radio station that is transmitting the stream without the name parameter.

Noisy interference on the radio

If there is noise interference when playing the radio and this is still present even when the radio is muted this can be for two reasons. This can happen with a wired Ethernet connection to the Raspberry Pi and is the source of the interference. This can be cured by connecting to a Wireless Network (WIFI) using a wireless adapter.
Another less common cause can be an inadequate power supply. See Power supply considerations on page 16.

Configuring a wireless adaptor

You will almost certainly want to configure a wireless adaptor for the radio instead of a wired network connection. Choose a wireless adapter that has been approved for the Raspberry PI. See the following link for approved Raspberry PI peripherals:
http://elinux.org/RPi_VerifiedPeripherals

Install the wireless adapter

Switch off the Raspberry PI and plug in the adaptor into one of the USB ports. Power the PI back on and log in. Check to see if your Wireless Adapter has been recognised by running the **lsusb** command.

```
pi@raspberrypi:~$ lsusb
Bus 001 Device 002: ID 0424:9512 Standard Microsystems Corp.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 003: ID 0424:ec00 Standard Microsystems Corp.
Bus 001 Device 004: ID 148f:5370 Ralink Technology, Corp. RT5370 Wireless
Adapter
```

The above shows a Ralink (Tenda) wireless adaptor but this will vary depending on the adapter that has been installed.

Configure the adaptor

The configuration is contained in the **/etc/network/interfaces** file as shown below

```
pi@raspberrypi:~$ cat /etc/network/interfaces
auto lo

iface lo inet loopback
iface eth0 inet dhcp

allow-hotplug ne  0
iface wlan0 inet manual
wpa-roam /etc/wpa_supplicant/wpa_supplicant.conf
iface default inet dhcp
```

You should not need to change this file. The file to be amended is shown on the line beginning with wpa-roam and is **/etc/wpa_supplicant/wpa_supplicant.conf**. Edit this file.
It will only contain a couple of lines.

```
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
```

Add the following after the above lines:

```
 network={
    ssid="YOUR_SSID"
    scan_ssid=1
    psk="YOUR_KEY"
    proto=RSN
    key_mgmt=WPA-PSK
    pairwise=CCMP
    auth_alg=OPEN
}
```

Substitute YOUR_SSID and YOUR_KEY with the actual SSID and key for your wireless router. The above configuration is for a router using WPA encryption. If your router is using the older WEP encryption then you will need to adapt the configuration to use WEP. See next section.

Explanation of the network fields

Field	Description
ssid	your WIFI (SSID) name
scan_ssid	A value of 1 means broadcast and value of 2 means a hidden SSID (Normally enter a value of 1)
psk	Your WIFI password
proto	Your choice of RSN or WPA. RSN is WP2 and WPA is WPA1. (most configurations are RSN)
key_mgmt	Either WPA-PSK or WPA-EAP (pre-shared or enterprise respectively)
pairwise	Either CCMP or TKIP (WPA2 or WPA1 respectively)
auth_alg	OPEN option is required for WPA and WPA2 (other option, SHARED & LEAP)

The only problem with the above configuration is that the **psk** key is in plain text and can be read by anyone who has access to the Raspberry PI. It is possible to increase security by generating a so-called passphrase with the **wpa_passphrase** command. For example if your **ssid** is *mywlan* and the WIFI password is *abcdef1234* then use the following command to generate the passphrase.

```
# wpa_passphrase mywlan abcdef1234
network={
        ssid="mywlan"
        #psk="abcdef1234"
        psk=53a566e0ccf03ec40b46e6ef4fc48b836e428fb0fd5e0df95187ba96e60ce7ce
}
```

Copy and paste the passphrase into the psk parameter into the **/etc/wpa_supplicant/wpa_supplicant.conf** file. Do not include any quotes around it.

Operating the wireless interface

If configured correctly the wireless adapter will start up when the Raspberry PI is rebooted.

The adaptor can be started and stopped with the following commands:

```
root@raspberrypi:/home/pi# ifup wlan0
```

and

```
root@raspberrypi:/home/pi# ifdown wlan0
```

To see what **SSIDs** are available run the iwlist command as shown in the following example:

```
root@raspberrypi:/home/pi# iwlist wlan0 scanning | grep ESSID
                    ESSID:"mywlan"
                    ESSID:"VGV751926F4B9"
                    ESSID:"prime"
                    ESSID:"Sitecom6A212C"
```

To display the IP address of the Wireless Adapter run the **ip addr** command:

```
# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP qlen 1000
    link/ether b8:27:eb:fc:46:15 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.11/24 brd 192.168.2.255 scope global eth0
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen
1000
    link/ether c8:3a:35:c8:64:cd brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.13/24 brd 192.168.2.255 scope global wlan0
```

Troubleshooting the wireless adapter

Problem – Starting the wireless adapter gives the following message:

```
# ifup wlan0
wpa_supplicant: /sbin/wpa_supplicant daemon failed to start
run-parts: /etc/network/if-pre-up.d/wpasupplicant exited with return code 1
Failed to connect to wpa_supplicant - wpa_ctrl_open: No such file or
directory
wpa_supplicant: /sbin/wpa_cli daemon failed to start
run-parts: /etc/network/if-up.d/wpasupplicant exited with return code 1
```

This is due to an incorrect **/etc/wpa_supplicant/wpa_supplicant.conf** file. The problem is due to an incorrect configuration. For example a space after the **ssid=** directive as shown below.

```
network={
    ssid= "homelan"
    scan_ssid=1
    psk="d762c954df"
    proto=RSN
    key_mgmt=WPA-PSK
    pairwise=CCMP
    auth_alg=OPEN
}
```

Solution: Correct the error and run the **ifup wlan0** command.

Problem: The following is seen:

```
root@raspberrypi:/home/pi# ifup wlan0
ifup: interface wlan0 already configured
```

Solution: This isn’t actually an error. Just run the **ifdown wlan0** command and retry the **ifup wlan0** command. It should then work.

Streaming to other devices using Icecast2

Introduction to Icecast

You may wish to play the output of the Radio through your PC speakers or a mobile device such as a tablet or telephone. This is possible to do this using **Icecast**. For more information on Icecast see the following Wikipedia article: http://en.wikipedia.org/wiki/Icecast .

Please also refer to *Intellectual Property, Copyright, and Streaming Media* on page 79.

Installing Icecast

Install **icecast2** as user root user

```
# apt-get install icecast2
```

The Icecast2 installation program will ask if you wish to configure Icecast2.
Answer '**yes**' to this. Configure Icecast as follows:

Icecast2 hostname: **localhost**
Icecast2 source password: **mympd**
Icecast2 relay password: **mympd**
Icecast2 administration password: **mympd**

It is important that you replace the default password 'hackme' with 'mympd' and that you leave the Icecast2 hostname as 'localhost'.
The installation program continues configuration. The icecast2 server will be started

```
Done Configuring icecast2..
insserv: warning: script 'ggpiod' missing LSB tags and overrides
Starting icecast2: Starting icecast2
Detaching from the console
icecast2.
```

Ignore the innserv warning message.

Change the mount point for **icecast** by editing the **/etc/icecast2/icecast.xml** configuration file. Find the following section in the file:

```
<!-- Only define a <mount> section if you want to use advanced options,
           like alternative usernames or passwords
<mount>
        <mount-name>/auth_example.ogg</mount-name>
```

Change the **mount-name** parameter to **/mpd** .

```
<mount>
        <mount-name>/mpd</mount-name>
```

Don't forget the **/** in front of **mpd** Save the **/etc/icecast2/icecast.xml** configuration file.

Create the mount point and make it world accessible:

```
# mkdir /mpd
```

Restart the icecast2 server:

```
# service icecast2 restart
Restarting icecast2: Starting icecast2
Detaching from the console
icecast2.
```

Edit the **/etc/mpd.conf** file and add the following lines to the end of the file:

```
# MPD Radio Stream
audio_output {
  type              "shout"
  name              "PI Radio MPD Stream"
  description       "MPD stream on Raspberry Pi Radio"
  host              "localhost"
  port              "8000"
  mount             "/mpd"
  password          "mympd"
  bitrate           "128"
  format            "44100:16:2"
  encoding          "mp3"
}
```

Save the **/etc/mpd.conf** file and restart MPD.

```
# service mpd restart
Stopping Music Player Daemon: mpd.
Starting Music Player Daemon: mpd.
```

Check that the PI Radio stream is enabled

```
# mpc outputs
Output 1 (My ALSA Device) is enabled
Output 2 (PI Radio MPD Stream) is enabled
```

Check that MPD has established a connection with the icecast2 server

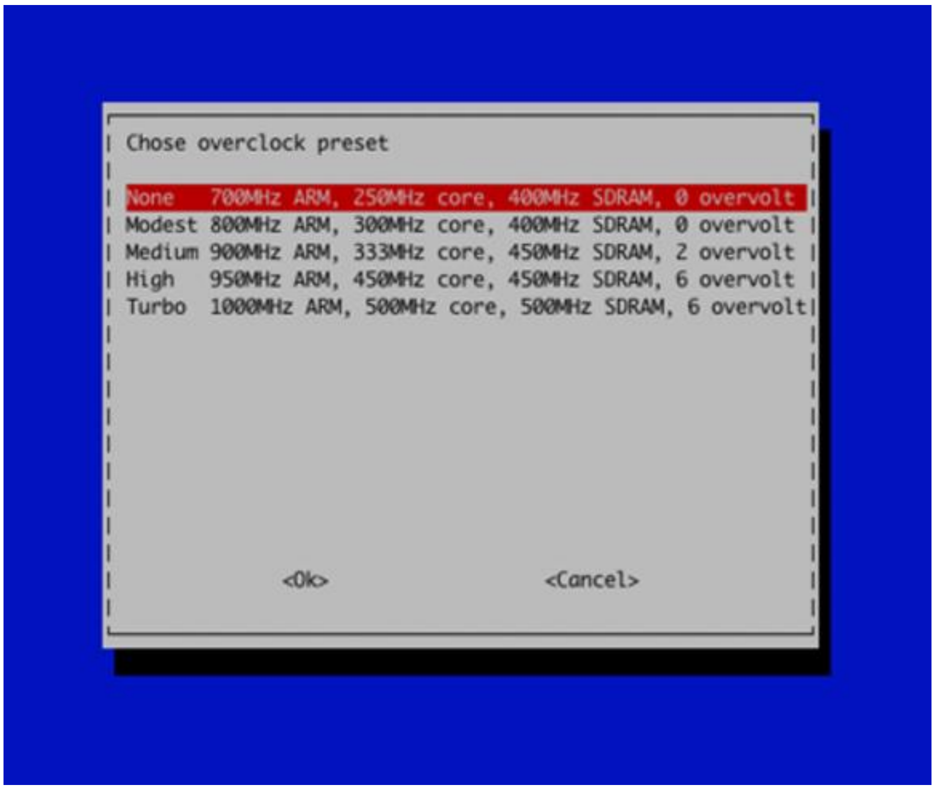
```
# netstat -tn | grep :8000
tcp        0      0 127.0.0.1:8000      127.0.0.1:38639         ESTABLISHED
tcp        0      0 127.0.0.1:38639     127.0.0.1:8000          ESTABLISHED
```

This completes the installation of Icecast2 however you may need to configure the clock speed.

Overclocking the Raspberry PI

It will almost certainly be necessary to over-clock the Raspberry PI to handle Icecast2 streaming using the **raspi-config** program. Medium over-clocking seems to be sufficient.

Run **raspi-config**. Select option 7 'Overclock'. The following screen will be displayed.



```
Chose overclock preset

None    700MHz ARM, 250MHz core, 400MHz SDRAM, 0 overvolt
Modest  800MHz ARM, 300MHz core, 400MHz SDRAM, 0 overvolt
Medium  900MHz ARM, 333MHz core, 450MHz SDRAM, 2 overvolt
High    950MHz ARM, 450MHz core, 450MHz SDRAM, 6 overvolt
Turbo   1000MHz ARM, 500MHz core, 500MHz SDRAM, 6 overvolt
```

Figure 28 Over-clocking the Raspberry PI

Select 'Medium' to start with. Reboot the Raspberry PI when prompted.

Re-test the radio with streaming switched on.

Icecast2 Operation

Note this section will change in the next release to allow the radio to have full control over the Icecast2 service.

Disable the Icecast2 service during reboot

When the radio is first switched on the Icecast2 streaming service will start automatically unless the service has been disabled. Disable the Icecast2 service with the following command.

```
# update-rc.d icecast2 disable
```

Don't disable the service if you want it always to be on by default.

Switching on streaming

Before you can listen to the streaming on the PC or mobile device it is necessary to start the Icecast2 streaming daemon. It must be switched on first.

There are three ways of doing this:

1. Switch on the Streaming in the Radio's options menu (Use this one)
2. Start the icecast2 service manually.
3. Enable Icecast2 during reboot

Using the radio options menu

Use the options menu (Press menu button three times). Step through the menu option using the Channel up/down buttons until "Streaming off" is displayed in the LCD display (assuming Icecast is installed). Press either Volume button and after a short delay the text should change to "Streaming on" in the LCD display. Press the menu button again to exit the options menu.

This starts the Icecast2 service. It also writes the word "on" or "off" to a file called **/var/lib/radiod/streaming**. This is file is used to enable or disable the Icecast streaming function at boot time.

Starting Icecast2 manually

Use the following command:

```
# service icecast2 start
```

To stop it again:

```
# service icecast2 start
```

Enabling Icecast2 at reboot time

It isn't necessary to enable the Icecast2 service at boot time as the radio program will start it depending on the contents of the **/var/lib/radiod/streaming** file. Should you wish to enable streaming at boot time then enable it with the following command.

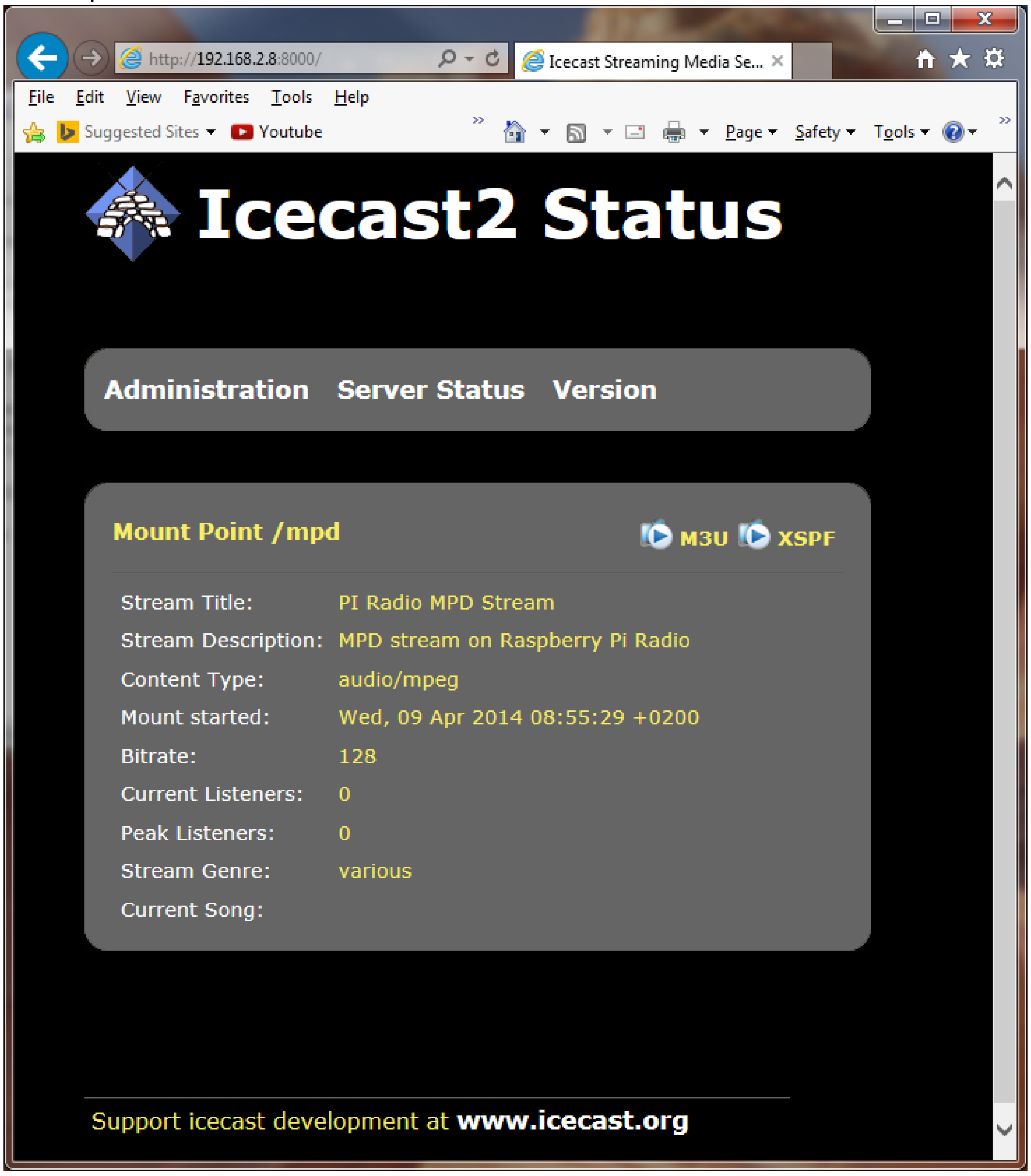
```
# update-rc.d icecast2 enable
```

Playing the Icecast stream on a Windows PC

To play the Icecast2 radio stream on a PC point your web browser at the IP address of the radio on port 8000. In the following example the IP address of the radio is 192.168.2.11. So this would be:

http:// 192.168.2.11:8000

The following screen should be displayed. If not continue to the troubleshooting guide at the end of this chapter:


http://192.168.2.8:8000/
Icecast Streaming Media Se...
File Edit View Favorites Tools Help
Suggested Sites
Youtube
Page Safety Tools
Icecast2 Status
Administration Server Status Version
M3U XSPF

Mount Point /mpd

```
Stream Title:         PI Radio MPD Stream
Stream Description:   MPD stream on Raspberry Pi Radio
Content Type:         audio/mpeg
Mount started:        Wed, 09 Apr 2014 08:55:29 +0200
Bitrate:              128
Current Listeners:    0
Peak Listeners:       0
Stream Genre:         various
Current Song:
```

Support icecast development at **www.icecast.org**

Figure 29 Icecast2 Status

Click on the **M3U** link to play the stream. This will launch your configured music player (For Windows PCs this is normally Windows Media Player.

The selected radio station or music track should be heard through the PC speakers.

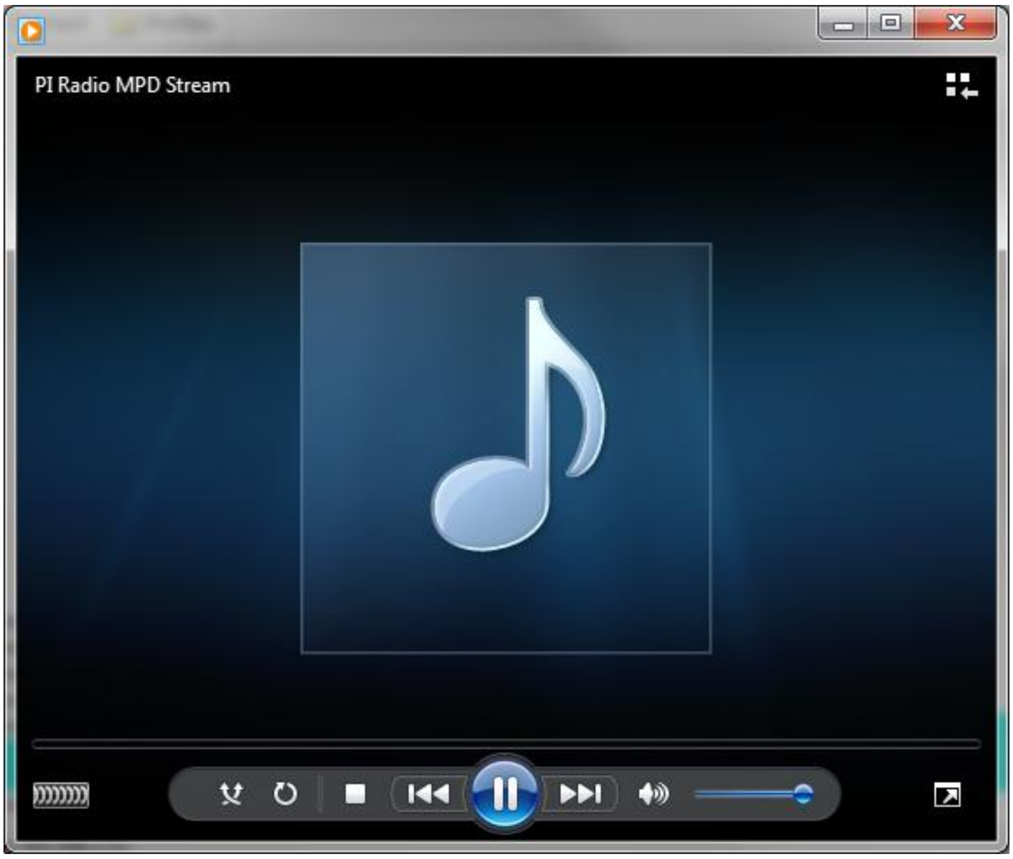


Figure 30 Windows media player

At this point you may wish to mute the sound from the radio itself. Use the radio Mute function. The playing stream will not be affected.

Playing the Icecast2 stream on an Apple IPad

This is exactly the same as playing the Icecast2 stream on a Windows PC.

1. Open the Safari browser.
2. Type in the Icecast2 URL. For example **http://192.168.2.11:8000**
3. Click the M3U button

This should open the iTunes Player and after a short time should start playing the radio stream.

Playing the Icecast2 stream on an Android device

1. Open your web browser
2. Type in the Icecast2 URL. For example **http://192.168.2.11:8000/mpd** (don't include .m3u)
3. When asked to "Complete action with" select your *Android System* then *Music player*

The Icecast stream should start playing. It is important <u>not</u> to key in **mpd.m3u** at the end of the URL. It must be **mpd** only.

Visual streaming indicator

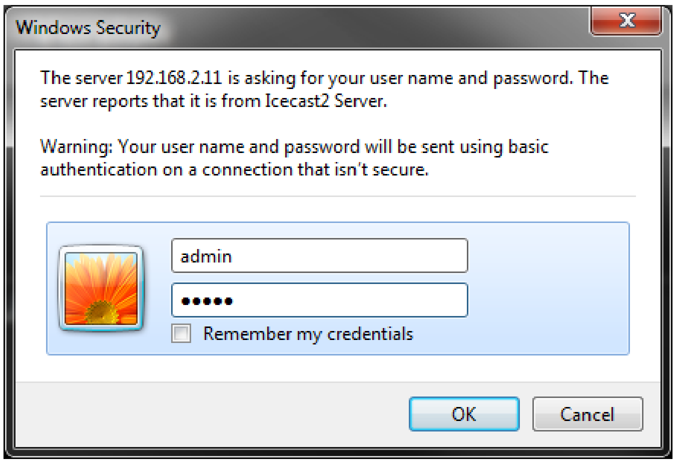
When streaming is switched on an asterix '*' character is displayed as a visual streaming indicator in the LCD display on the Raspberry PI radio. When the '*' character is displayed this indicates that the Icecast2 streaming is switched on.

For the four line 20 character display the visual indicator is displayed after the time on the first line.
09:26 02/05/2014 *

For the two line by 16 character display there isn't the room to do this so it is displayed after the Volume or Mute message on the second line.
Volume 75 * or Sound muted *

Administration mode

In the "Icecast Status" screen there is an Administration tab. Click on the Administration tab. The following screen will be displayed. Enter username 'admin' and the password 'mympd'.



Windows Security

The server 192.168.2.11 is asking for your user name and password. The server reports that it is from Icecast2 Server.

Warning: Your user name and password will be sent using basic authentication on a connection that isn't secure.

admin

●●●●●

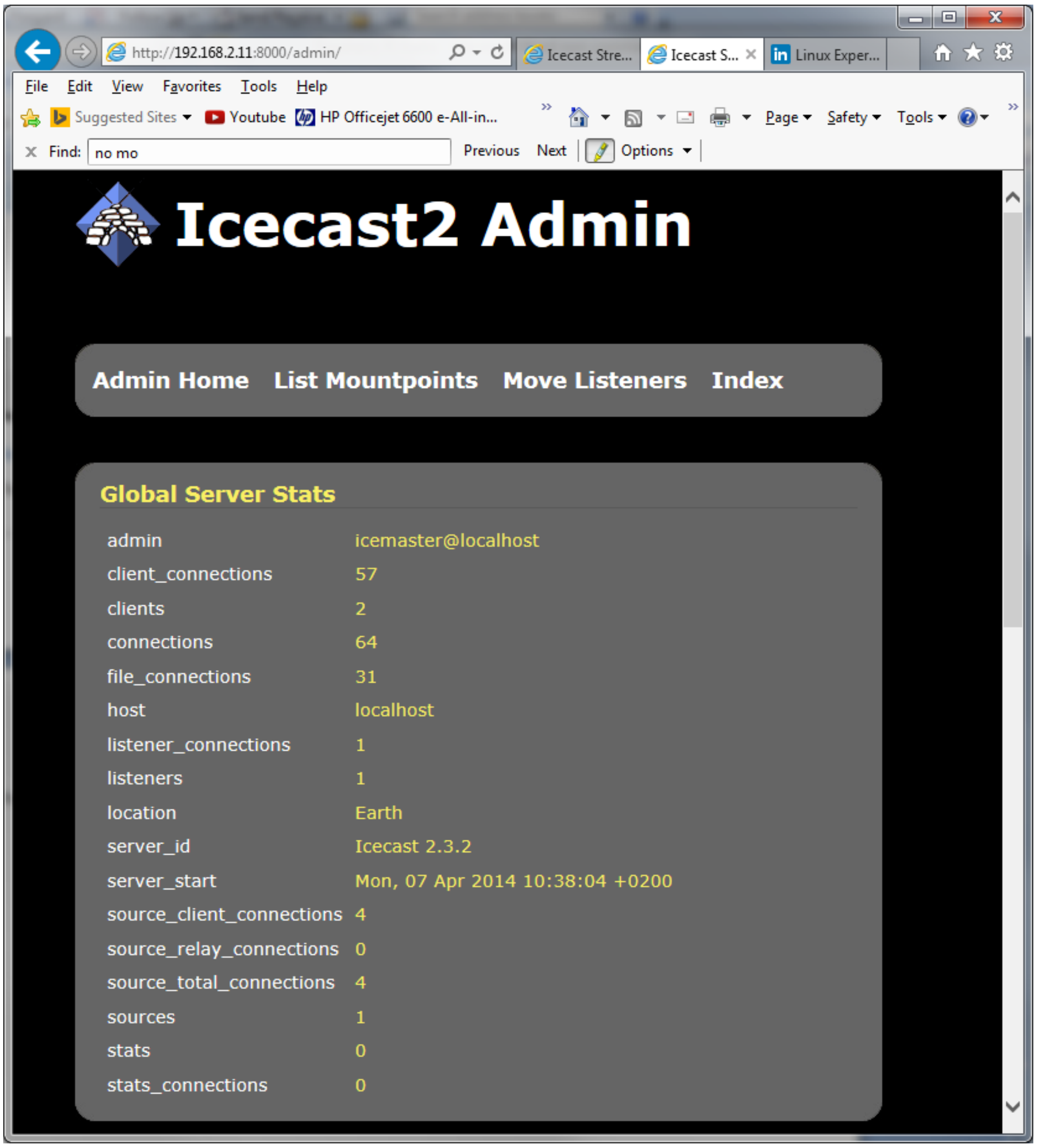
☐ Remember my credentials

Figure 31 Icecast admin login

The icecast2 administration page will be displayed.

It has two parts. The Global Server statistics and mount point information.

See next page:


http://192.168.2.11:8000/admin/
Icecast Stre...
Icecast S...
Linux Exper...
File Edit View Favorites Tools Help
Suggested Sites
Youtube
HP Officejet 6600 e-All-in...
Page Safety Tools
Find: no mo
Previous Next Options
Icecast2 Admin
Admin Home List Mountpoints Move Listeners Index

Global Server Stats

admin	icemaster@localhost
client_connections	57
clients	2
connections	64
file_connections	31
host	localhost
listener_connections	1
listeners	1
location	Earth
server_id	Icecast 2.3.2
server_start	Mon, 07 Apr 2014 10:38:04 +0200
source_client_connections	4
source_relay_connections	0
source_total_connections	4
sources	1
stats	0
stats_connections	0

Figure 32 Icecast Global Server Status

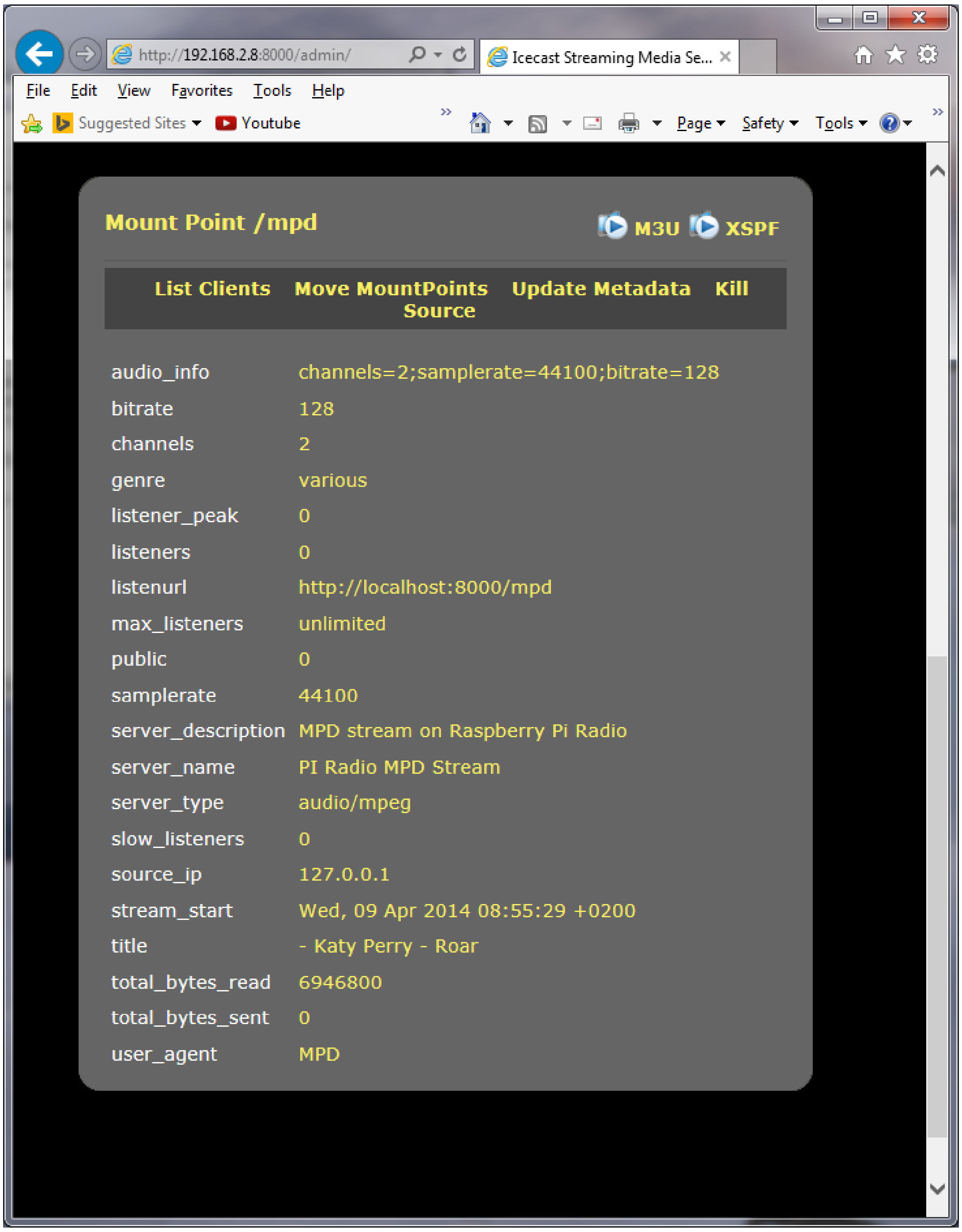
It is beyond the scope of this manual to describe **Icecast2** administration. See the following site for further support. That said you should not need to change anything.

http://www.icecast.org/

The above site contains latest software updates, information, documentation and support forums.

See next page:

The second part (scroll down to display). This will display the **/mpd** mount point information and the currently playing track. This screen isn't automatically refreshed so you will need to refresh it if you want to display the name of any new track or station.


http://192.168.2.8:8000/admin/
Icecast Streaming Media Se...
File Edit View Favorites Tools Help
Suggested Sites
Youtube
Page Safety Tools
M3U XSPF
List Clients Move MountPoints Update Metadata Kill Source

Mount Point /mpd

audio_info	channels=2;samplerate=44100;bitrate=128
bitrate	128
channels	2
genre	various
listener_peak	0
listeners	0
listenurl	http://localhost:8000/mpd
max_listeners	unlimited
public	0
samplerate	44100
server_description	MPD stream on Raspberry Pi Radio
server_name	PI Radio MPD Stream
server_type	audio/mpeg
slow_listeners	0
source_ip	127.0.0.1
stream_start	Wed, 09 Apr 2014 08:55:29 +0200
title	- Katy Perry - Roar
total_bytes_read	6946800
total_bytes_sent	0
user_agent	MPD

Figure 33 Icecast2 Mount point information

Troubleshooting Icecast2

Help for general problems with icecast2 can be found on the forums at http://www.icecast.org/

Icecast2 has two log files in the **/var/log/icecast2** directory namely **access.log** and **error.log**. The error log may give a clue as to the problem.

Below is a simulated error caused by mis-configuring the shoutcast entry in /etc/mpd.conf file. Here the hostname 'piradio' has been configured in the **/etc/mpd.conf** shoutcast entry instead of 'localhost'.

```
$ tail -f /var/log/mpd/mpd.log
Apr 07 10:43 : output: Failed to open "PI Radio MPD Stream" [shout]: problem
opening connection to shout server piradio:8000: Couldn't connect
```

Problem - Icecast streaming page says it can't be displayed.

Possible causes:

- The icecast service is not running on the radio.
  - Start it either from the Radio options menu (Streaming on) or run **sudo service icecast2 start** on the Rasperry PI and retry.
- Incorrect IP address or missing port number in the URL.
  - See Icecast2 *Operation* on page 71

Problem – No Mount Point displayed

Possible causes:

- This is mostly due to a mis-match in the MPD configuration and the Icecast2 configuration.
  - The icecast configuration is file is **/etc/icecast2/icecast.xml** . Make sure that all of the passwords are set to 'mympd'. The password 'hackme' will not work.
- There is no /mpd directory or the permissions are incorrect.
  - Check that the **/mpd** directory exists and that the permissions are set to 777. See Installing *Icecast* on page 68.

Problem - Cannot play the stream on my Android device

There are a number of Icecast players which can be downloaded onto Android and play Icecast2 streams across the network without problem. However the usual Android System Music player should work. The most likely cause of this problem is keying in an incorrect URL (Maybe adding .m3u to the end). See *Playing the Icecast2 stream on an Android device* on page 73.

Problem – Music keeps stopping or is intermittent

This is difficult to give a definitive answer to this problem. It must be remembered that running MPD and Icecast2 together on a Raspberry PI is pushing the Raspberry PI to its limits. It can also depend on your network or the PC you are using. Personal experience showed no problem playing a stream on PC with a wired network connection however a Laptop connected over a wireless network did not work well. Trying to play two or more devices on the MPD/Icast2 stream is also likely to result in poor results.

Try over-clocking the Raspberry PI using the **raspi-config** program. Medium over-clocking seems to be sufficient. See Overclocking the Raspberry PI on page 70. The icecast streaming facility is a fun thing to try out but if it doesn't work properly or is causing you stress; switch the streaming facility off.

Frequently asked questions (FAQs)

What is the login name and password?

The default login name is: pi
The password is: raspberry

Why are the radio stations not in the order that they were defined?

Playlists are loaded by the radio daemon in alphabetic order using the playlist name.
When loading an individual playlist, MPD loads the stations in the order that they are defined in each individual playlist.

It helps greatly to group stations of the same type into a single playlist. For example group all BBC radio stations into a single playlist.

The only way to get all of the radio stations in the order that you define them is to define a single playlist, for example **myplaylist**:

```
(myplaylist)
#
# United Kingdom
[BBC Radio 1] http://bbc.co.uk/radio/listen/live/r1.asx
[BBC Radio 2] http://bbc.co.uk/radio/listen/live/r2.asx
[BBC Radio 3] http://bbc.co.uk/radio/listen/live/r3.asx
:
:
[RAIradio3] http://www.listenlive.eu/rai3.m3u
```

This will produce a single playlist called **myplaylist.pls** with the stations loading in the order that they have been defined in the **/var/lib/radiod/stationlist** file. Make sure there are no blank lines between station definitions otherwise this terminates the playlist. All remaining stations will end up in there own single playlist file.

Why are some station names not being displayed in the web interface?

The reason for this is that some stations don't send the name with the stream. If you run the **mpc playlists** command you will see that some radio stations shows only the station URL and not the name:

```
$ mpc playlist
RAIradio2
:
BBC Radio 4 extra
http://icestreaming.rai.it/1.mp3
BBC Radio 3
BBC Radio 6
BBC Radio 5 live
```

The only way around this is to complain directly to the radio station to ask them to amend their stream to include the station name and title details. The only way reason that the staion name is seen with the radio program is that it picks up the names out of the station list file.
The snoopy web interface can't do this however.

Licences

The software and documentation for this project is released under the GNU General Public Licence.

The GNU General Public License (GNU GPL or GPL) is the most widely used free software license, which guarantees end users (individuals, organizations, companies) the freedoms to use, study, share (copy), and modify the software. Software that ensures that these rights are retained is called free software. The license was originally written by Richard Stallman of the Free Software Foundation (FSF) for the GNU project.

The GPL grants the recipients of a computer program the rights of the Free Software Definition and uses *copyleft* to ensure the freedoms are preserved whenever the work is distributed, even when the work is changed or added to. The GPL is a *copyleft* license, which means that derived works can only be distributed under the same license terms. This is in distinction to permissive free software licenses, of which the BSD licenses are the standard examples. GPL was the first *copyleft* license for general use.

See http://www.gnu.org/licenses/#GPL for further information on the GNU General Public License.

Intellectual Property, Copyright, and Streaming Media

This is an unbelievably complex subject. The author is not a lawyer and can not offer any legal advice on this subject. If you decide to stream your music content or relay a radio station stream back out to the internet then you should seek legal advice.

In general Radio stations are providing a stream to promote their radio station. As media providers they should have arrangements in place to make the content that they provide is legally streamed across the Internet but not all do. The question is it legal to listen (or view) such content is a complex one and subject to local and international laws and which vary considerably.

If you implement Icecast to re-stream content then provided that this is not streamed back out to the Internet then one would think that you will not encounter any problems (but you never know).

If you stream music tracks or relay radio stations back out onto the internet then almost certainly you will be infringing a copyright law or intellectual property rights somewhere. The penalties for such an infringement can be severe.

WARNING: YOU USE THE ICECAST STREAMING IN THIS PROJECT AT YOUR OWN RISK ESPECIALLY IF YOU MAKE THE STREAM CONTENT AVAILABLE ACROSS THE INTERNET, EVEN IF YOU ARE JUST RELAYING AN EXISTING MEDIA STREAM, LEGAL OR OTHERWISE, BACK OUT TO THE INTERNET.

Also see the Disclaimer on page 80.

Disclaimer

Technical support

Technical support is on a voluntary basis by e-mail only at bob@bobrathbone.com. Before asking for support, please first consult the troubleshooting section on page 60. I will always respond to e-mails requesting help and will never ignore them. I only ask that you do the same (i.e. Did my suggestions help or not?). Be sure to provide the following information:

- What have you built (Adafruit or normal LCD variants) and which program you are running?
- A clear description of the fault.
- What you have already done to locate the problem?
- Is anything displayed on the LCD?
- Did you run the test programs?
- Switch on DEBUG logging as described on page 41, run the program and include the log file.
- Did you vary from the procedure in the manual or add any other software?

Please note that support for general Raspberry PI problems is not provided. Only issues relating to the Radio software will be investigated.

For general Raspberry PI support see the following site:

http://www.raspberrypi.org/forums/

For support on Music Player Daemon issues see the help pages at the following link:

http://www.musicpd.org/

For issues relating to Icecast2 streaming see http://www.icecast.org

Acknowledgements

My thanks to Matt Hawkins for the original LCD screen driver routines. It made the job of writing the *lcd_class.py* much easier.

The original instructions on how to use Rotary Encoders came from an excellent article by Guy Carpenter. See:
http://guy.carpenter.id.au/gaugette/2013/01/14/rotary-encoder-library-for-the-raspberry-pi/
His ideas were used in the *rotary_class.py* code.

To Adafruit Industries for their excellent LCD plate and I2C code. See http://www.adafruit.com.

To Steffen Müller for his article on Streaming audio with MPD and Icecast2 on Raspberry Pi.
See http://www.t3node.com/blog/streaming-audio-with-mpd-and-icecast2-on-raspberry-pi/

To contributors such as Alan Broad who supplied photos of the Lego example of the radio plus code contribution. Also to Mike Whittaker for his contribution on how to drive the USB speaker set. Other contributors such as James Rydell (Old Zenith Radio)

To all constructors of this project who have sent in photos of their radio's and their ideas for improvement and the many appreciative e-mails that I have received from them.

Glossary

AAC	Advanced Audio Coding
ASX	Advanced Stream Redirector
CGI	Common Gate Interface – Executable Server Side scripts
CIFS	Common Internet File System
DHCP	Dynamic Host Configuration Protocol
GPIO	General Purpose IO (On the Raspberry PI)
I2C	Industry standard serial interface (Philips) using data and clock signals
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
LCD	Liquid Crystal Display
M3U	MPEG3 URL
MPC	Command line client for MPD
MPEG	Moving Picture Experts Group
MPEG3	Music encoding standard from MPEG
NAS	Network Attached Storage
NFS	Network File System
NTP	Network Time Protocol
MPD	Music Player Daemon
PC	Personal Computer
PID	Process ID
PLS	MPEG Playlist File (as used by Winamp)
RSS	Really Simple Syndication – Web feed usually containing news items
SD	San Disk Memory Card commonly found in cameras and Smartphones
SSID	An SSID is the public name of a wireless network.

URL	Universal Resource Locator (A link to a Web page for example)
USB	Universal Serial Bus
WEP	Wired Equivalent Privacy (WEP) is a security algorithm considered less secure than WPA
WIFI	Wireless Network using the 802.11 Wireless Network protocol
WPA	Wi-Fi Protected Access (WPA) and Wi-Fi Protected Access II (WPA2)